Energy Efficient Reconfigurable SRAM Using Data-Dependency

by

Chuhong Duan

B.S., Electrical Engineering, University of Virginia (2013)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2015

Author .....
Department of Electrical Engineering and Computer Science
Aug 28, 2015

Certified by .....
Anantha P. Chandrakasan
Joseph F. and Nancy P. Keithley Professor of Electrical Engineering
Thesis Supervisor

Accepted by .....
Leslie A. Kolodziejski
Chair, Department Committee on Graduate Students

Energy Efficient Reconfigurable SRAM Using Data-Dependency

by

Chuhong Duan

Submitted to the Department of Electrical Engineering and Computer Science
on Aug 28, 2015, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

The binary values processed and stored at the intermediary stages of an algorithm are often highly correlated. Motivated in part by this observation and the ever-increasing challenge of power density for Integrated Circuit (IC) systems, a novel reconfigurable memory framework is proposed in this thesis which builds upon traditional low power techniques such as voltage scaling in order to achieve up to 31% power savings for targeted applications. The general strategy underlying the presented low power memory innovation is to leverage global and local data correlation in order to make predictions so that the overall switching activity on the read bit-lines of the Static Random Access Memory (SRAM) is reduced with minimal area overhead. Additionally, multiple prediction schemes are incorporated into this framework wherein statistical data features are used to optimally configure each column of the proposed SRAM. Analysis tools for developing this type of reconfigurable low-power memories are provided.

An example reconfigurable CP SRAM adhering to the proposed framework is presented which includes the novel designs of a 10-transistor (10T) bit-cell, a prediction-based conditional pre-charge scheme, and a column-wise reconfigurable dual prediction mode architecture. A 16kbit SRAM incorporating these innovations is implemented in a test chip using a 28nm FD-SOI CMOS process. Using post-layout simulations, the proposed SRAM is found to provide 14%-20%, 4%, and 31% reductions in read power as compared with a conventional 8T SRAM for three targeted applications: the coefficient SRAMs in a sparse Fast Fourier Transform (sFFT) implementation, the Support Vector Machine (SVM) weights SRAM in an objection detection system, and the Motion Estimation (ME) reference pixel SRAM in a video coding system, respectively.

Thesis Supervisor: Anantha P. Chandrakasan
Title: Joseph F. and Nancy P. Keithley Professor of Electrical Engineering

Acknowledgments

This thesis and the work it entailed would not have been possible without the guidance and support of my advisor, Professor Anantha Chandrakasan. Anantha, thank you for giving me the opportunity to join your group and providing me with incredible resources to learn and grow while undergoing this challenging yet exciting long-distance run. Aside from the technical lessons I learned from you, your passion and dedication to education and the circuit community have been an inspiration to me. I am deeply grateful for the enlightening discussions and warm encouragements you've shared throughout.

I am grateful to have had the opportunity to learn from some of the most brilliant, dedicated, and down-to-earth people I have ever met while in Ananthagroup. I would like to thank my colleagues and friends in Ananthagroup for their generous support in solving technical problems of all sorts. Specifically, I'd like to thank Avishek and Yildiz for welcoming me to the SRAM family and helping me ramp up when I first joined. They taught me countless things and were always willing to share more. I feel fortunate to have had them as mentors. I'd like to thank Arun, Mehul, Michael, and Phil for always answering my calls for help and generously sharing their insights and experience. I'd like to thank my current or former group mates: Bonnie, Chiraag, Dina, Dongsuk, Frank, Gilad, Georgios, Hyung-Min, Nachiket, Nathan, Omid, Preet, Priyanka, Rahul, Sirma, and Sungjae. I'd also like to thank the kindest and simply best admin, Margaret, for all of her assistance. It is a privilege to be a member of this group and I have greatly enjoyed my residency thus far.

I would like to thank a few wonderful individuals I have been fortunate enough to collaborate with during this thesis. I'd like to thank Mahmut for being an amazing mentor during my summer internship. I learned a lot from that experience and from the many other colleagues at NVIDIA. I'd like to thank Amr and Omid for the meetings, email discussions, and efforts in helping me understand the applications essential to this thesis. I'd like to thank Avishek, Arun, and Mehul for patiently analyzing and debugging detailed circuit diagrams with me in the lab, on the plane, at

ISSCC, or remotely over the phone. I'd like to thank Tarek for his help in preparing the final manuscript and matlab programming. I would also like to thank DARPA and NSF for funding support and STMicroelectronics for their generous support with chip fabrication.

To all my friends, thank you for your care and support and for encouraging me to be who I am. I will forever cherish all the joy and laughter we have shared.

I would like to thank Tarek, my best friend and my soul mate, for being the most effective dose of caffeine in getting me to work and the greatest support when things go wrong. I admire you and have learned so much from you. Thank you for being so generous and wonderful.

Finally, I'd be remiss if I didn't thank my Mom Weilai Zhong, Dad Bin Duan, and family for their endless support and unconditional love. Thank you for believing in me and providing me with the best to be who I am today. It is your love that keeps me moving forward and I am forever grateful to you.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Continuous process scaling, driven by the exponential growth of the semiconductor industry as famously predicted by Moore's law [18], has led to innovation in the area of integrated circuit (IC) design and has primarily resulted in increased functionality, lower fabrication and production costs, and continuously shrinking form factors. This trend of transistor feature size reduction, however, has also introduced challenges including increased power density and heat dissipation which are undesirable to small, energy-constrained mobile devices that rely on limited battery and/or self-harvested energy.

1.1.1 Designing for energy constrained applications and systems

The power consumed by a digital circuit is dominated by two components: dynamic power ($P_{dynamic}$) and leakage power ($P_{leakage}$). Specifically, dynamic power consumption is attributed to capacitive logic switching, i.e. the charging and discharging of capacitors. When a circuit node goes from low (0) to high (1), current is drawn from the power

supply to fully charge the capacitor(s) attached to that node. Dynamic power is typically given by

$$P_{dynamic} = \alpha_{0\to1} C V_{dd}^2 f \tag{1.1}$$

where $\alpha_{0\to1}$ is the activity factor, C is the switching capacitance, V_{dd} is the supply voltage, and f is the clock frequency. On the other hand, leakage power is attributed to the undesired leakage current that flows between power sources, intermediary nodes, and ground as a result of the fact that transistors are not completely off in their off state. Leakage power is typically given by

$$P_{leakage} = I_{leakage} V_{dd} \tag{1.2}$$

where $I_{leakage}$ is the leakage current.

While the switching capacitance C scales proportionately with process technology, digital circuits at advanced nodes are designed to have more functionality per chip, more gates and therefore more switching nodes per unit area, and faster clock frequencies. As a result, the ever-increasing challenge is power density for IC systems. Additionally, leakage power in scaled processes is increased due to, for example, lower threshold voltages (V_T) and thinner gate oxides. For the reasons stated above, the exploration of low-power techniques for IC systems remains an active research area.

Embedded Static Random Access Memories (SRAMs) are critical components in modern system-on-chips (SoCs). As the capabilities of many digital electronic devices continue to improve at roughly exponential rates, the need for both large and low-power on-chip storage grows in parallel [4]. In IC implementations for various applications, SRAMs occupy a disproportionate amount of both the total die area and the total power consumption [37]. While recent progress has allowed the digital blocks of a system to be operated at lower supply voltages, robust SRAM operations still require a high supply voltage in order to guarantee reliability under worst-case

process, temperature, and voltage conditions. Figure 1-1 shows the scaling trend of SRAM bit-cell sizes and SRAM power supply voltages V_{MIN} (minimum V_{dd} at which an SRAM works reliably) over process nodes. Observe that SRAM supply voltage scales much slower than the bit-cell size due to the stability challenges mentioned above. Consequently, SRAM has recently become the bottleneck of further power reduction in many systems and thus necessitates creative energy and area-efficient solutions.

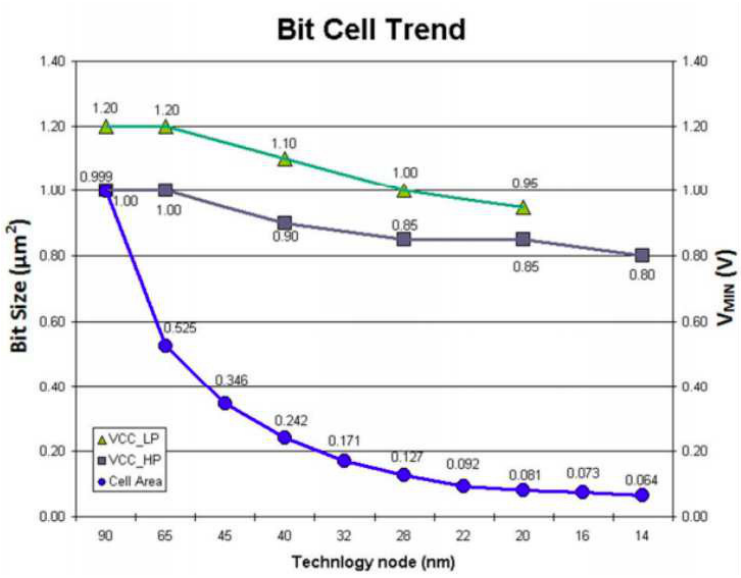

Figure 1-1: Bit-cell and V_{MIN} scaling trend of SRAM from major semiconductor manufacturers. Figure adapted from ISSCC 2014 Trends.

1.1.2 Application specific considerations

In order to develop a new dimension of energy savings, application-specific data features have been explored in addition to conventional techniques such as voltage scaling, read-/write-assist circuitry, and variation-tolerant sensing schemes. Highly correlated data, defined in [26] as data with repeated or similar values in consecutive cycles or more specifically with low switching activity $\alpha_{0\to1}$, has been leveraged to help memory

make predictions, reduce bit-line switching, and ultimately save energy wasted on reading redundant and/or predictable information. Existing data-dependent designs, however, are limited by their narrow applicability as well as significant overhead in area and/or latency. In this work, we propose an alternative 10-transistor (10T) bit-cell and a novel conditional pre-charge column circuit to save read power while simultaneously minimizing the overhead introduced by generating predictions. In addition, we develop a framework for *reconfigurable* memory in which data features are used to select an optimal prediction scheme per column of the proposed SRAM, making it well-suited for various energy-constrained applications with a wide range of data features.

1.2 Summary of contributions

This thesis focuses on the design of energy-efficient, data-dependent SRAMs equipped with reconfigurable prediction schemes and the methods by which application data statistics may be utilized to identify optimal configurations. The key contributions of this work are twofold. First, we propose a general framework and the associated analysis tools for developing reconfigurable low-power memories that take into account application-specific information provided by software. Second, an example reconfigurable CP SRAM adhering to the proposed framework is presented which includes the novel designs of a 10T bit-cell, a prediction-based conditional pre-charge scheme, and a column-wise reconfigurable dual prediction mode architecture. Incorporating these innovations, a 16kbit SRAM test chip is developed in a 28nm Fully-Depleted Silicon-on-Insulator (FD-SOI) Complementary Metal-Oxide Semiconductor (CMOS) process. Using its post-layout simulations, the proposed SRAM is found to provide 14%-20%, 4%, and 31% reductions in read power as compared with a conventional 8-transistor (8T) SRAM for three targeted applications: the coefficient SRAMs in a

sparse Fast Fourier Transform (sFFT) implementation, the Support Vector Machine (SVM) weights SRAM in an objection detection system, and the Motion Estimation (ME) reference pixel SRAM in a video coding system, respectively.

Chapter 2

A survey of low power SRAMs

Key metrics of on-chip SRAM designs include power and/or energy, performance, area, and robustness. These metrics often imply critical design tradeoffs for which careful analysis and creative thinking may lead to an acceptable balance by taking into account the targeted application. This thesis focuses in particular on the design of low-power SRAMs for energy-constrained systems using this approach.

The total power consumption of SRAM consists primarily of power incurred during read and write accesses and leakage power wasted during data retention. This thesis emphasizes on SRAMs active operations for which average dynamic power, corresponding to per unit time energy consumed for charging and discharging capacitive loads, is regarded as the critical metric. The majority of SRAM dynamic power dissipation occurs during the switching of its highly capacitive bit-lines. In particular, the power consumed by bit-line switching can be written as

$$P = \alpha_{0\to1} C_{BL} V_{dd} (\Delta V) f, \tag{2.1}$$

where $\alpha_{0\to1}$ is the activity factor, C_{BL} is the bit-line capacitance, V_{dd} is the supply voltage, ΔV is the minimum amount of voltage development required on the bit-line for the sensing circuit to correctly resolve the read value, and f is the frequency of

operation.

This thesis considers applications in which the total number of data accesses is high and read operations occur much more frequently than write operations. Hence, our design effort is primarily spent on the reduction of dynamic power dissipated on the read bit-lines.

2.1 Overview of low power techniques

The conventional 6-transistor (6T) SRAM is the most common type of on-chip memory due to its merit of high density [12]. However, in its most primitive form, 6T is inherently energy-inefficient. Power is consumed on its bit-lines during every read operation irrespective of the specific data being read. In this sense, 6T SRAM has 'dynamic' read operations in that every data evaluation is contingent on a bit-line pre-charge activity which always consumes a considerable amount of power. This implies that the combined $\alpha_{0\to1}$ in Eq. 2.1 corresponding to both bit-lines is at its maximum ($= 1$) for 6T read operations.

In the literature, much work has focused on reducing power via voltage scaling, i.e. to lower the supply voltage V_{dd} or V_{min} as it is sometimes referred to in SRAMs. By lowering V_{dd} both leakage and dynamic power are saved [5]. However, operating SRAM at low voltages is generally challenging in that it requires a significant amount of design effort to deal with issues such as increased soft error rate, degraded read static noise margin, and reduced write margin due to the exponential impact of intensified V_T variations (e.g., [34, 23]). To guarantee SRAM's reliability and robustness under all operating conditions is a critical task and a research area of its own. We refer readers to [24] for a rigorous and complete development of this subject. Other approaches, for example minimizing ΔV by designing low-offset or offset-compensated sense amplifiers (e.g., [30]), have been undertaken. However, due to increased device variations at

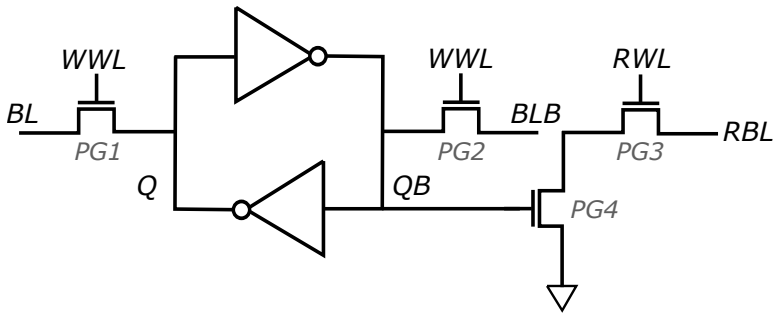

Figure 2-1: The conventional 8T bit-cell.

reduced V_{dd}, worst-case ΔV approaches V_{dd} in value.

2.2 The data dependent approach

As a consequence of the previously discussed limitations associated with voltage scaling, SRAM designers have looked into new ways of reducing memory power. One such way is to minimize switching activity $\alpha_{0\to1}$ of bit-lines using or introducing correlation in stored data. Data stored in an SRAM often has particular properties that can be leveraged for application-specific designs. An example of an application-specific SRAM is presented in [8]. This work recognizes the inherent data dependency of 8T SRAM, a bit-cell of which is depicted in Fig. 2-1. Referring to this figure, 8T SRAM's pre-charged read bit-line `RBL` is only discharged when reading a 0 (or a 1 if the single read port is connected to `Q`). By using majority logic and data-bit inverting, this design maximizes the number of 1s stored in the SRAM resulting in minimized total power consumption. This scheme works well for heavily skewed data. However, when the percentage of 1s is close to 50%, 8T SRAM consumes at least half of the read power of a 6T SRAM.

New bit-cells as well as architectural designs have helped in leveraging data correlation. For example, [26] and [7] respectively designed a prediction-based 10T SRAM and a priority-based 6T/8T hybrid SRAM to maximize total power savings in

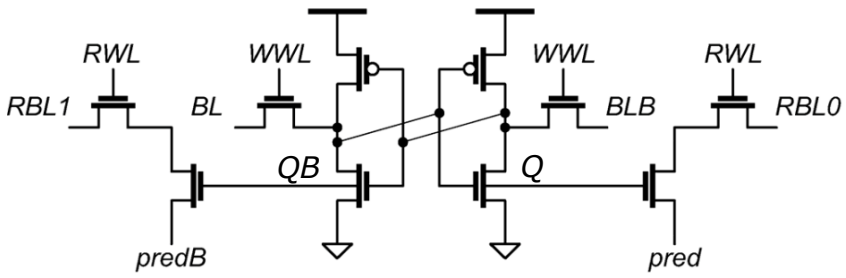

Figure 2-2: PB-RBSA bit-cell. Figure adapted from [26].

real-time video applications. The prediction-based reduced bit-line switching activity (PB-RBSA) SRAM design, as presented in [26], reduces bit-line switching activity by using correlation present in video data and a bit-cell topology that uses a bit-wise prediction provided by a separate arithmetic averaging block. As illustrated by the PB-RBSA bit-cell diagram in Fig. 2-2, the footer nodes of the two read ports are not connected to ground but two prediction lines `pred` and `predB`. With a correct prediction, i.e. `pred` = `Q` and `predB` = `QB`, both read bit-lines `RBL0` and `RBL1` stay at V_{dd} during a read evaluation. A major drawback of this design is the highly capacitive prediction lines `pred` and `predB` that span the entire height of the array. Therefore, driving and changing predictions creates a significant power overhead and introduces critical tradeoffs between the prediction accuracy and the power consumption of prediction lines.

Alternative SRAM models may also benefit from utilizing data correlation. For instance, the work in [21] eliminates the pre-charge phase entirely by using a 10T bit-cell with a statically driven read bit-line. A depiction of this bit-cell design is provided in Fig. 2-3. When reading the same or similar data between cycles, this *static* SRAM has minimal switching on its read bit-lines. However, both latency and area are major concerns of this approach.

In Table 2.2 we compare our approach against the previously mentioned relevant data-dependent memory models. As is evident in our discussion this far, previous

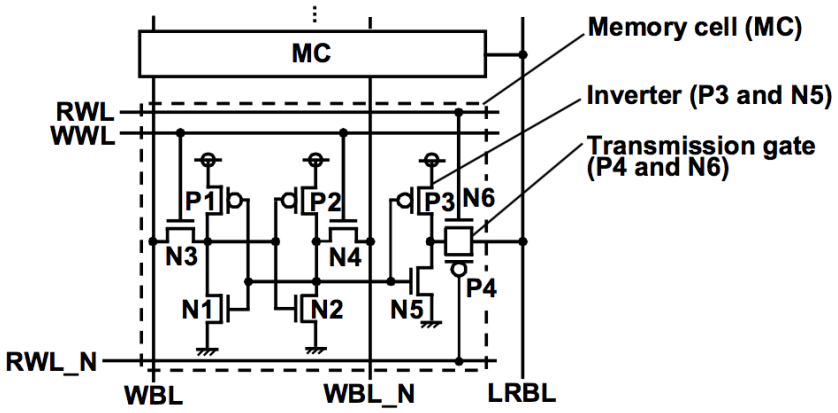

Figure 2-3: 10T non-precharge-type two-port memory cell. Figure adapted from [21].

Table 2.1: Comparison with relevant data-dependent SRAM models

Work	Design Idea	Targeted Application(s)
Noguchi '07 [21]	A non-precharge 10T SRAM for reduction in bit-line switching activity	video applications
Fujiwara '08 [8]	An 8T SRAM using majority logic and data-bit reordering for reduction in bit-line switching activity	video applications
Chang '11 [7]	A priority-based 6T/8T hybrid SRAM for aggressive voltage scaling	video applications
Sinangil '14 [26]	A 10T PB-RBSA SRAM using output prediction for reduction in bit-line switching activity	video coding
This work	A conditional pre-charge 10T SRAM using reconfigurable prediction schemes for reduction in bit-line switching activity	sFFT, object detection, video coding

work has identified opportunity in a data-dependent SRAM approach. However, room for improvement exists in that no one has fully developed a generic model that would work well across various applications. Toward this end, this thesis aims to leverage these ideas by incorporating a wide range of data features into the design of a single SRAM. Both circuit- and architectural-level techniques are explored in order to realize dynamic power savings with minimal area overhead. In Section 3.3.3, we compare our proposed SRAM against the models mentioned in this section.

Chapter 3

Energy-efficient reconfigurable memory

3.1 A general framework

In this chapter, we present our reconfigurable memory framework in which application specific information such as data statistics are generally obtained via software in order to dynamically configure our memory hardware. Two statistical measures are repeatedly considered in the analysis in this thesis, namely the *global correlation* measure ρ and the *local correlation* measure $\alpha_{0\to1}$, equivalently referred to as the switching activity factor. The global correlation measure shows how biased a data sequence is in its bit values. It is generated by

$$\rho \triangleq \frac{M}{N}, \tag{3.1}$$

where M is the number of occurrences of the majority bit in a given data sequence of length N. For example, 1 is the majority bit in the sequence 1111111100 since there are eight 1s out of the total ten bits. In this case, $M = 8$, $N = 10$, and therefore $\rho = 0.8$ or 80%. On the other hand, the local correlation measure shows how often a

$0 \rightarrow 1$ bit-level change occurs in a given data set. It is generated by

$$\alpha_{0\rightarrow 1} \triangleq \frac{\text{number of } 0 \rightarrow 1 \text{ transitions}}{N-1}. \tag{3.2}$$

With these definitions in place, we comment that the local and global correlation measures respectively satisfy $\alpha_{0\rightarrow 1} \in [0, 0.5]$ and $\rho \in [0.5, 1]$ while not all $(\rho, \alpha_{0\rightarrow 1})$ pairs in $[0.5, 1] \times [0, 0.5]$ are obtainable since the measures are correlated themselves. We refer to correlated data in what follows as a data sequence with high global and/or local correlation, i.e. a large ρ and/or a small $\alpha_{0\rightarrow 1}$.

As illustrated by Fig. 3-1, statistical parameters such as ρ and $\alpha_{0\rightarrow 1}$ are passed from software to the prediction generation block, i.e. the *predictor*, which communicates directly with the memory array. The software program specifically takes the profiled data set for a targeted application and passes to the predictor a number of computed statistical measures. Equipped with this information as well as other inputs from the memory, the predictor identifies the optimal prediction mode from which prediction values for each memory read operation are generated and provided to the memory unit. In this sense, the memory in this framework is reconfigurable since a number of prediction modes are available for selection. Equipped with this infrastructure and knowledge of application-specific statistics, the proposed memory can support low-power data access for various classes of applications with distinct data features.

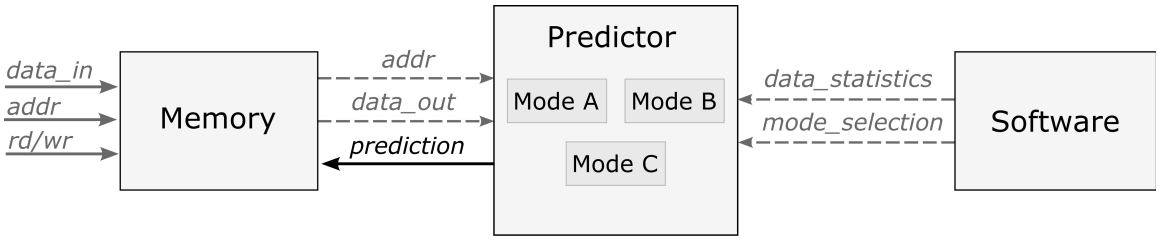

Figure 3-1: An illustration emphasizing the communication between software and memory hardware via the predictor in the proposed reconfigurable memory framework.

This thesis in part focuses on the exploration and design of a compact architecture

which endorses reconfigurability as previously described. In addition, pertinent questions involving the selection of appropriate prediction schemes and the identification of sufficient statistics to discern which scheme is best suited to a particular application are addressed. In the remainder of this chapter, we present our design at different hierarchical levels and address the above questions by means of simulation using a number of novel and existing memory designs. We conclude this chapter with an example SRAM adhering to the presented framework which constitutes our final design.

3.2 Conditional pre-charge SRAM

3.2.1 Proposed bit-cell design

Figure 3-2 shows our proposed bit-cell design. The 10T bit-cell depicted subsumes a standard 6T bit-cell, i.e. two cross-coupled inverters and two access NMOS transistors, and two additional read ports. Read operations are decoupled from write operations in this memory cell since separate word lines (RWL and WWL) and bit-line pairs (RBL/RBLB and BL/BLB) are used for read and write control and accesses, respectively. Equipped with separated read ports, the 10T bit-cell is exempt from read stability issues as is the conventional 8T memory cell. Additionally, it may be used to construct two-port SRAMs wherein simultaneous read and write bit-line accesses are allowed. This design avoids the prediction line power overhead observed in the 10T bit-cell design in [26] by grounding the sources of the two bottom read port devices. Alternatively, prediction signals are passed to the memory via pre-charge circuitry as is discussed in Section 3.2.2.

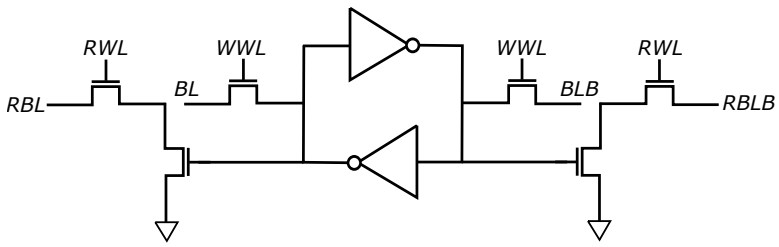

Figure 3-2: The proposed 10T bit-cell consisting of two cross-coupled inverters, two access NMOS transistors for the write bit-lines BL and BLB, and two read ports connected to the read bit-lines RBL and RBLB.

3.2.2 Conditional pre-charge

The column architecture of the proposed 10T memory array is provided in Fig. 3-3. Write operations are handled using the conventional 6T portion of the bit-cell. In particular, BL and BLB are driven to the desired values of the input and its complement, and then, with the WWL asserted, values on these bit-lines get passed to the bit-cell storage nodes `Q` and `QB` and overwrite their original values. Read operations in this design selectively utilize one of the read ports depending on the current prediction value. The general idea is to always pre-charge the bit-line on the side that is predicted to store a 0, a scheme we henceforth refer to as *conditional pre-charge*. In particular, two PMOS transistors (`P1` and `P2`) in the pre-charge circuit are controlled by a pair of prediction signals, specifically, by the prediction `pred` and its complement `predB`. When the pre-charge enable signal `prechB` is asserted, one of the read bit-lines is charged through the turned-on PMOS stack while the other read bit-line's connection to the power supply is cut off. A pair of crossed-coupled NMOS transistors (CNN) keeps the unselected read bit-line low while the selected one is pre-charged high. Hereon, we refer to this design as the Conditional Pre-charge SRAM (CP SRAM).

Figure 3-4 illustrates two example cases for a read operation. Suppose the bit-cell nodes have the values `Q` $= 0$ and `QB` $= 1$. Figure 3-4(a) depicts a correct prediction,

i.e. `pred` = 0. In this case, RBL is charged high in the pre-charge phase while RBLB is kept low. During evaluation, the discharge path is open thus RBL stays high for the next read cycle. Figure 3-4(b) depicts an incorrect prediction. In this case, pre-charged RBLB discharges through the turned-on PMOS transistors during evaluation. Making correct predictions for the CP SRAM leads to reduced total pre-charge activity and consequently decreases the total energy consumed on the bit-lines.

It should be noted that the above arguments are valid if off-state leakage current is small as compared with the total on-state current. This is generally the case for regular threshold voltage (regular V_T) devices in technologies such as FD-SOI [15]. If leakage current is significant and the evaluation phase is sufficiently long, the pre-charged read bit-line will leak from V_{dd} to a low logic value even when the prediction is correct and the associated PMOS transistors are turned off. This would result in an incorrect read output. In the development of a prototype test chip using regular V_T devices in a FD-SOI technology, we further ensure the functionality of the presented read scheme by using fixed-width timing controls as is discussed in Chapter 4. Secondary effects including leakage current are considered in the post-layout simulations presented in Section 4.2.

As expected, the prediction sequence directly affects the amount of power savings achievable in prediction-based SRAMs. Arithmetic averaging is a suitable scheme for the PB-RBSA SRAM since changing prediction from cycle to cycle causes significant power overhead and therefore should be balanced by using a larger prediction generation window. For CP SRAM, however, updating the prediction value has a negligible cost overhead due to the embedded prediction devices `P3` and `P4` in the pre-charge circuitry. Making a prediction complementary to the current output invariably causes switching in the following pre-charge phase and in the evaluation phase if the updated prediction is incorrect (e.g., due to signal noise). In order to minimize these types of unnecessary switchings when data is sufficiently correlated, we propose the Previous-Read-Data

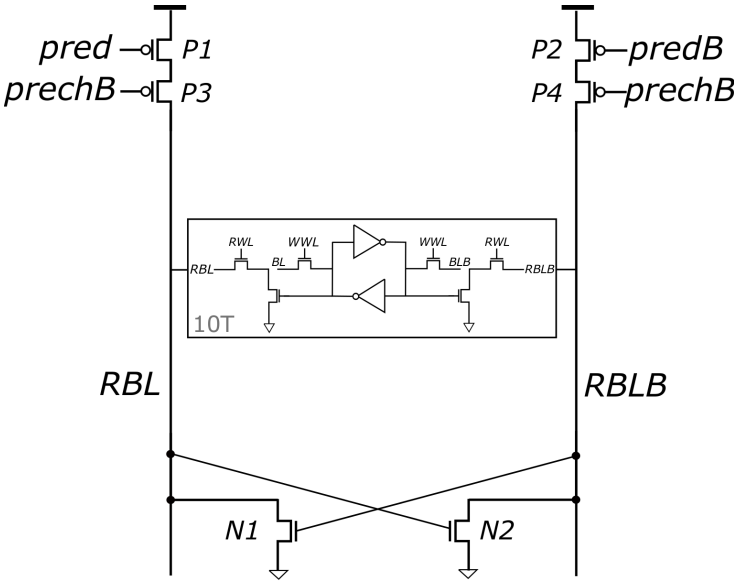

Figure 3-3: Column architecture of the proposed Conditional Pre-charge SRAM (CP SRAM). One of the read bit-lines is pulled high depending on the current prediction values (`pred`, `predB`) via the associated pair of pre-charge PMOS transistors. The other read bit-line is kept low by one of the cross-coupled NMOS transistors `N1` and `N2`.

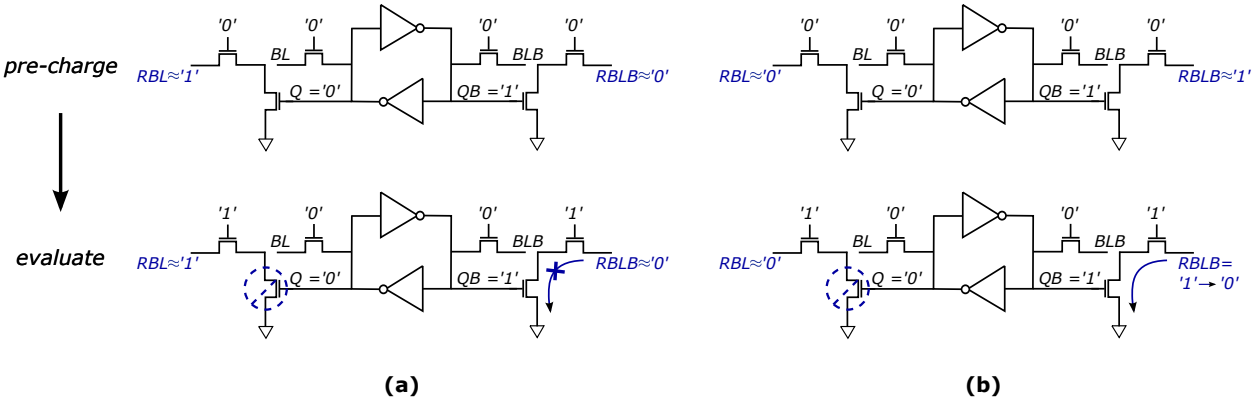

Figure 3-4: (a) Correct and (b) incorrect prediction cases during a read operation with the CP SRAM. The conditionally pre-charged read bit-line remains high if the prediction is correct and is discharged to low if the prediction is wrong.

(PRD) prediction scheme where the data value currently being accessed is used to generate the next prediction. Our analysis shows that for a broad range of data profiles, PRD is a better option as compared with arithmetic averaging in terms of area, simplicity, and power savings with correlated data. Consistent with this remark,

a simulation-based analysis of various prediction schemes is presented in Section 3.3.1. There are, of course, corner cases where PRD does not result in power saving, e.g., when $\alpha_{0\to1} = 0.5$. These cases will be examined in detail and handled by selecting a different prediction mode of the reconfigurable SRAM. Finally, it should be noted that more complex and application-specific prediction generation designs can easily be incorporated into the framework. However, since this work serves as a proof of concept and demonstration of a simple generic system, we henceforth focus our attention to only PRD and arithmetic averaging with variable length average window sizes.

3.3 Evaluation of memory models and prediction schemes

We proceed to evaluate the tradeoffs amongst various memory models and their associated performance capabilities for different data sets using a custom dynamic power simulation program wherein the total read power consumption of application-specific and synthetic data alike is computed for each model. In high density arrays, bit-line switching activity accounts for the majority of total power consumption during read accesses [26]. Therefore, for the sake of tractability, we focus our evaluation to power consumption on read bit-lines. The custom program design specifically allows for flexibility in the sense of parameterized bit-line and prediction-line capacitances as well as bit-line voltage swing.

We respectively propose and consider the following designs:

(i) *CP-SRAM* : conditional pre-charge SRAM

(ii) *8T* : conventional 8T SRAM wherein the bottom gate of the read port is connected to `Q`

(iii) *static* : non-precharge static SRAM

(iv) *PB-RBSA* : prediction-based reduced bit-line switching activity SRAM

where (ii)-(iv) are as previously mentioned and listed here for completeness.

The justification of these designs for energy-constrained systems is twofold. First, by using separated read port(s), resulting in high read stability, these memories have the potential to support aggressive voltage scaling. Second, the read power consumption of each design depends critically upon data statistics such as $\alpha_{0\to1}$, as was discussed previously. Further power savings can be achieved by leveraging the data-dependent capability of each memory design thus motivating a careful analysis of these dependencies via the custom program.

The simulation program takes as an input SRAM data in one of three ways: specified by the user, synthesized using correlation parameters such as $\alpha_{0\to1}$ and ρ, or profiled from an application. For a given data array, the simulator first populates the corresponding prediction array for each of the prediction-based SRAMs (CP-SRAM and PB-RBSA) for the prediction schemes indicated by the user. Then, for each memory type, it identifies the power-incurring switching activities associated with reading data out the array in a sequential order. Bit-line power, as described in Eq. 2.1, is incurred on the bit-line(s) during pre-charge activity for 8T, PB-RBSA and CP-SRAM and during *1-after-0* data accesses for static SRAM. For PB-RBSA, switching the `pred` and `predB` lines contributes additional power consumption, which we henceforth regard as similar in cost to the power incurred on the bit-lines for simplicity. Loosely speaking, this implies that $C_{PL} \approx C_{BL}$. Lastly, adhering to Eq. 2.1, the total power consumption for each column of the memories considered, parameterized using the bit-line capacitances (C_{BL} and C_{PL}) and bit-line voltage swing (ΔV), is computed.

Sensing networks are a critical peripheral circuit in an SRAM. The choice of utilizing small signal sensing by employing Sense Amplifiers (SA) to amplify small signal variations on the bit-lines into full swing output signal, or large signal sensing directly affects the minimum value of ΔV. Other contributing factors include the

implementation technology as well as the process, voltage and temperature variation effects. Generally, a ΔV value of 0.1V is achievable with $V_{dd} = 1$V. However, considering worst case process variations when operated at a lower V_{dd} and/or the power overhead of offset compensation circuit, we assume that the ΔV associated with small signal sensing is approximately $\frac{2}{5}$ of the ΔV associated with large signal sensing which is itself approximately equal to V_{dd}. The bit-line(s) of PB-RBSA and 8T SRAM are often evaluated with SAs whereas the bit-line of static SRAM and the prediction lines of PB-RBSA require full-swing operations. We summarize this assumption as

$$\Delta V \approx \begin{cases} V_{dd}, & \text{for static, PB-RBSA prediction-line} \\ \frac{2}{5}V_{dd}, & \text{for 8T, PB-RBSA} \end{cases} . \tag{3.3}$$

We consider CP-SRAM with both small and large signal sensing networks. We next comment on the read power simulations for each of the three data types under the assumptions established in this subsection.

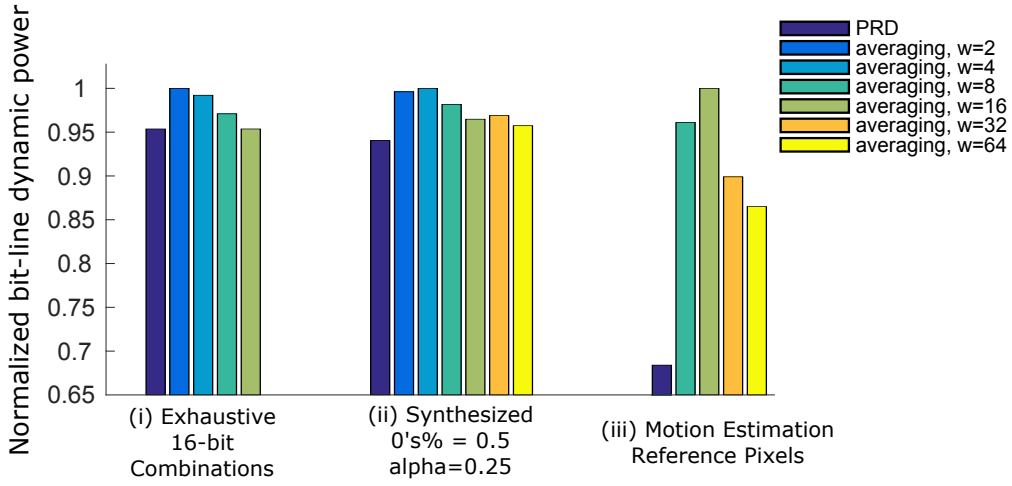

Figure 3-5: Normalized read bit-line power consumption of CP-SRAM using prediction scheme PRD *vs* arithmetic averaging for various average window sizes (w). Three data sets are used: (i) 2^{16} 16-bit binary number combinations, (ii) synthetic data with $\alpha_{0\to1} = 0.25$ and $\rho = 0.5$, and (iii) 1080×1920 highly compressed ME reference frame pixels.

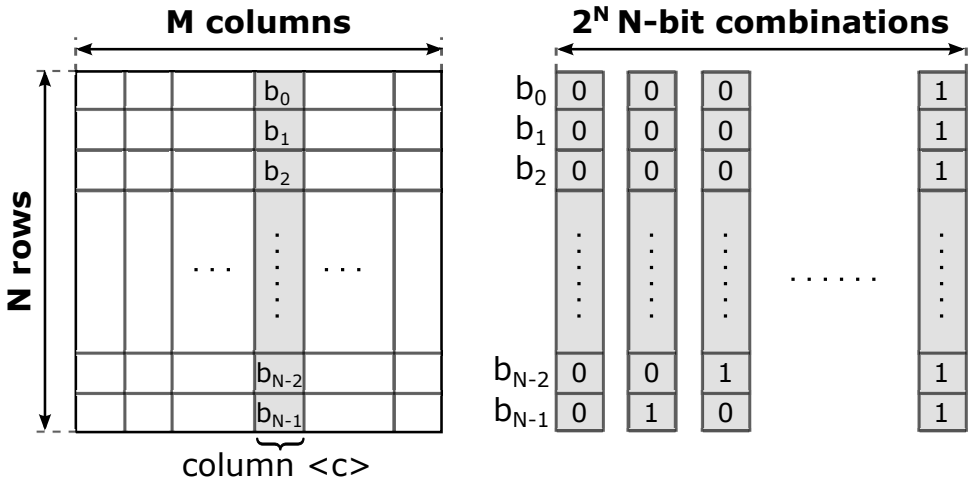

Figure 3-6: Organization of the input data set (i). The exhaustive collection of 2^N N-bit binary number combinations cover all possible patterns of the data stored in column <c> in an SRAM of size $N \times M$.

3.3.1 Prediction scheme

The normalized total read power consumption of CP-SRAM is depicted in Fig. 3-5 for both PRD and arithmetic averaging prediction schemes where the average window size $w = 2^k$ ranges from $k = 1$ to $k = 6$. The results reported correspond to three input data sets: (i) an exhaustive collection of 2^N N-bit binary number combinations for $N = 16$, (ii) a randomly synthesized data set with statistics $\alpha_{0\to1} = 0.25$ and $\rho = 0.5$, and (iii) 1080×1920 highly compressed Motion Estimation (ME) reference pixels extracted using the scheme discussed in Section 5.2.2. In regard to data set (i), the organization of the data set into memory is illustrated by Fig. 3-6 for an SRAM of size $N \times M$. Referring to an arbitrary column <c>, the bit values are read sequentially, i.e. $b_0 \to b_{N-1}$. As is indicated on the right, the exhaustive nature of (i) ensures that all possible patterns of data that can be stored in a column are considered. Furthermore, we comment that the mapping of an N-bit binary sequence to a $(\rho, \alpha_{0\to1})$ pair is not unique and so this approach facilitates the analysis of all possible $(\rho, \alpha_{0\to1})$ pairs achievable by a binary sequence of length N. For the purpose of demonstration and consistent with a wide range of values of N, the results displayed in Fig. 3-5 use

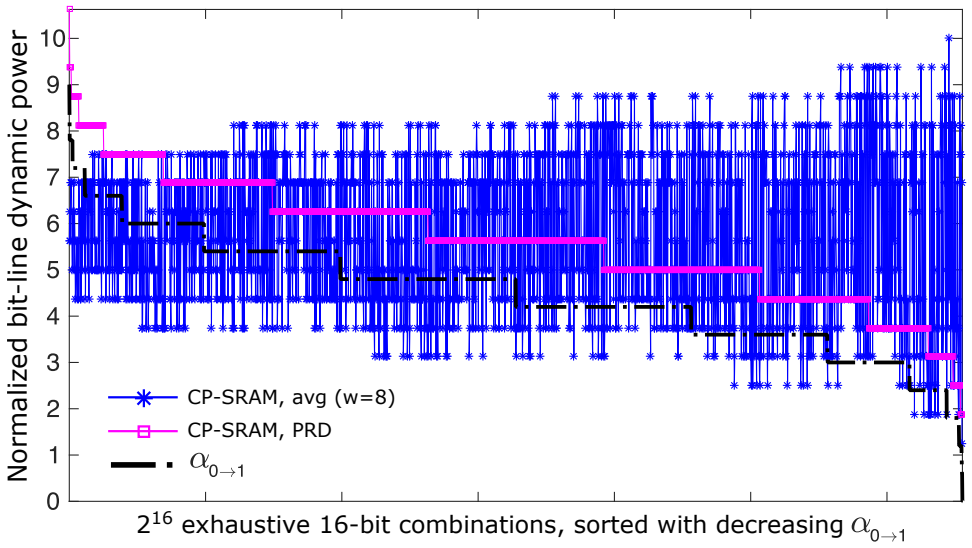

Figure 3-7: Normalized read bit-line power consumption of CP-SRAM for every 16-bit binary number combination for prediction scheme PRD and arithmetic averaging ($w = 8$). The indexing corresponds to decreasing $\alpha_{0\to1}$ for the exhaustive data set.

$N = 16$. We remark that our analysis and conclusions do not rely critically upon the specific value of N and in this sense we proceed taking $N = 16$ hereon for consistency.

Observe in Fig. 3-5 that only the applicable average window sizes are simulated for data sets (i) and (iii). Nonetheless, common to all three input data sets, PRD-based prediction results in the lowest total bit-line power consumption. This advantage of PRD is most evident for ME reference pixels which, on average, have a higher local correlation measure (a lower $\alpha_{0\to1}$) than the 16-bit binary number combinations.

In order to compare the performance of these prediction schemes more rigorously, the read power consumption is computed for each data sequence in the exhaustive set of 16-bit binary number combinations. The normalized power consumption of these sequences, sorted according to decreasing switching activity, shows that power consumption of CP-SRAM with PRD decreases monotonically with $\alpha_{0\to1}$ whereas arithmetic averaging is not smoothly related to $\alpha_{0\to1}$ (Fig. 3-7). For low values of $\alpha_{0\to1}$ (toward the right-hand side of the plot) PRD incurs less power for most cases and on average as compared with arithmetic averaging. This result is consistent with

our previous discussion, in particular that PRD is a better scheme in terms of power savings when the data is locally correlated which manifest itself as a low $\alpha_{0\to1}$ value. Subsequently, we proceed in our presentation by pairing the CP SRAM design with the PRD prediction scheme and henceforth refer to their joint use as CPSRAM.

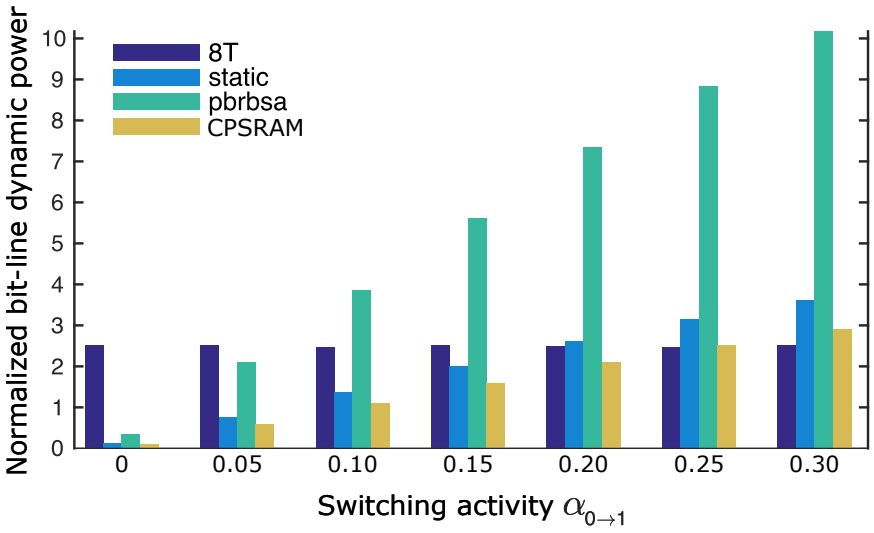

Figure 3-8: Normalized read bit-line and prediction-line power of various memory models for synthesized data sets with various switching activity factors $\alpha_{0\to1}$. All data sets have $\rho = 0.5$.

3.3.2 Power dependency on data statistics

As was previously mentioned, a built-in subroutine of the custom simulation program allows synthetic SRAM data to be generated for a prespecified pair (ρ, $\alpha_{0\to1}$). This capability allows us to explore the data-dependent features of each memory design in great detail. For instance, as shown in Fig. 3-8, the read power of both prediction-based and static SRAM is seen to scale appropriately with increasing $\alpha_{0\to1}$ while ρ, or specifically here the percentage of 0s of each data set, is held constant at 0.5. The read power of 8T, however, remains relatively constant with respect to switching activity. Figure 3-9 plots the read power of the selected memory models using data

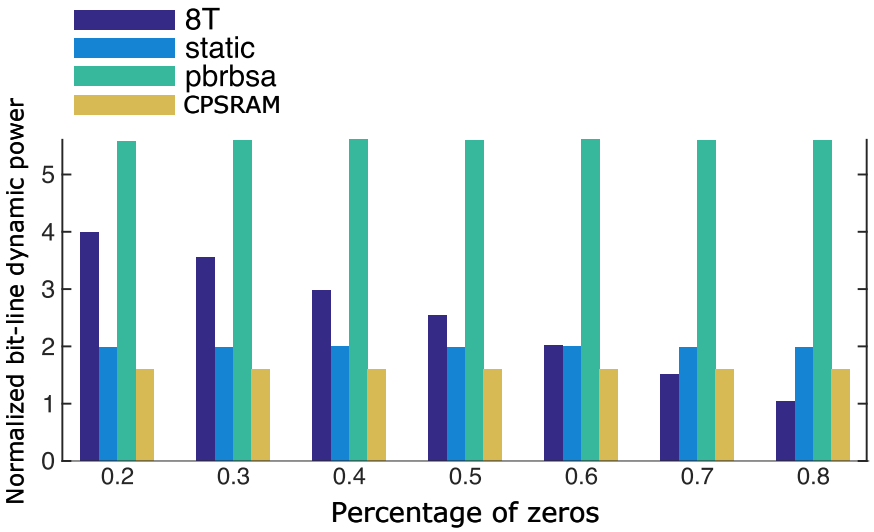

Figure 3-9: Normalized read bit-line and prediction-line power of various memory models for synthesized data sets with various percentages of 0s. All data sets have $\alpha_{0\to1} = 0.15$.

sets with constant $\alpha_{0\to1} = 0.15$ and increasing percentage of 0s. Observe that 8T's read power decreases as the number of 0s in the data set increases while the others' remain relatively constant.

3.3.3 Model evaluation

Pixel values from a 1080×1920 video frame, extracted from the 'Park Scene' video sequence belonging to the Joint Collaborative Team on Video Coding (JCT-VC) common test conditions [3], have been pre-profiled into an array structure, in mimicry of the block-based retrieval pattern found in High Efficiency Video Coding (HEVC) motion search. This is further discussed in Section 5.2.2. Figure 3-10 illustrates the normalized read power consumption of the video frame pixels for all four memory models. Referring to this figure, CPSRAM with both small (a) and large (b) signal sensing networks are considered. In particular, (a) assumes that ΔV(8T, PB-RBSA) : ΔV(CPSRAM) : ΔV(static, PB-RBSA prediction-line) = 1 : 1 : 2.5, i.e. the minimum

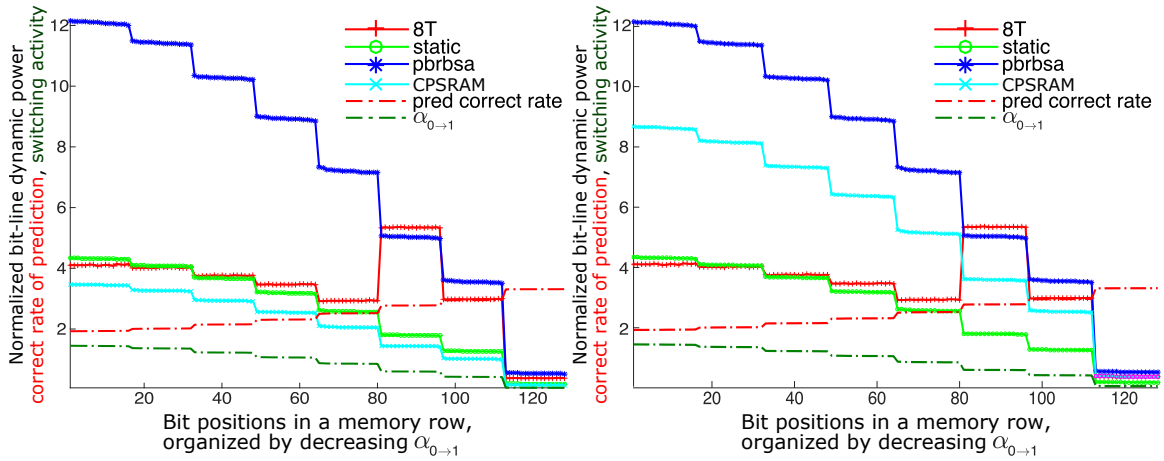

Figure 3-10: Normalized read bit-line and prediction-line power of various memory models for pixels in a highly-compressed HEVC ME reference frame. Bit positions are organized by decreasing $\alpha_{0\to1}$. Voltage swing ratio ΔV(8T, PB-RBSA) : ΔV(CPSRAM) : ΔV(static, PB-RBSA prediction-line) = (a) 1 : 1 : 2.5 and (b) 1 : 2.5 : 2.5.

amount of voltage development on the bit-line(s) of 8T, PB-RBSA, and CPSRAM is $\frac{2}{5}$ of that on the static SRAM read bit-line and the PB-RBSA prediction-lines. This assumption is only valid if 8T, CPSRAM, and PB-RBSA SRAMs are designed with small signal sensing circuits. On the other hand, if CPSRAM makes use of large signal sensing, then the second ratio of ΔV(8T, PB-RBSA) : ΔV(CPSRAM) : ΔV(static, PB-RBSA prediction-line) = 1 : 2.5 : 2.5 should be used for power computation as is the case in (b).

Observe from (b) that static SRAM is strictly better than CPSRAM in terms of bit-line power consumption when their switching costs are assumed to be equal. Loosely speaking, for any data profile CPSRAM has greater or equal $\alpha_{0\to1}$ as compared with static SRAM. However, a key disadvantage of static SRAM is that it is considerably slower in read operations as compared to the other SRAMs considered. Since the targeted applications in this thesis require real-time speeds, we proceed without further consideration of static SRAM. Correct resolution of read data in static SRAM requires

full bit-line voltage swing whereas the penalty of switching CPSRAM's bit-lines can be reduced by using sense amplifiers as shown in (a). We again call attention to the fact that the assumed ΔV ratios here are conservative. The actual amount of power that is saved by using sense amplifiers is often insignificant due to the large bit-line capacitance and number of bit-cells per bit-lines and therefore increased SA sensitivity to process variation, device mismatches, etc. [29, 31, 39]. Additionally, adding sense amplifiers to the CPSRAM design causes significant overhead in area. For the obvious reasons, we proceed with large signal sensing for CPSRAM.

For each column in Fig. 3-10(b), either 8T or CPSRAM has the lowest read power consumption depending on the data statistics of that column. For the specific case shown (average $\rho \approx 0.58$), CPSRAM consumes less power for columns with $\alpha_{0\to1} < 0.126$. This observation suggests that combining these two memory models into the design of a dual prediction mode SRAM would help reduce total power. Note that reading from an 8T SRAM is equivalent to reading from a CP SRAM using a *constant* prediction scheme, i.e. to always predict 0 or 1 for each column depending on which value constitutes the majority and consistently pre-charge the bit-line associated with the 0 prediction. In fact, CP SRAM with this bit-wise constant prediction scheme introduces the majority logic described in [8] to each column and would perform better than a data agnostic 8T SRAM. Thus, the hardware complexity associated with integrating two distinct SRAM models can be reduced by combining the two prediction schemes (PRD and constant) into the design of a single reconfigurable CP SRAM.

3.4 Proposed reconfigurable memory design

3.4.1 Two modes of read operation

In order to make accurate and low-power memory predictions for various data statistics, we propose in this thesis a conditional pre-charge SRAM with reconfigurable prediction modes wherein one of the two previously discussed prediction modes (PRD and constant) is selected for each SRAM column based on that column's data statistics. By independently operating each column in the preferred mode, i.e. finding and connecting the lowest points on the simulated power curves in Fig. 3-10 (b), we can achieve the maximum total read power savings for various applications and systems.

To this end, Fig. 3-11 shows an example decision tree on the left and an equivalent decision plane on the right. The depicted tree illustrates the procedure by which prediction modes are selected for each memory column. The presented decision tree is simplified in anticipation of the discussions in Chapter 5 surrounding three example applications and how, for each of them, the preferable SRAM configuration is identified in accordance with the decision branches. Referring to the figure, the red and blue coloring respectively indicate PRD and constant mode decisions. If the column's input data has a low global correlation, i.e. ρ is in the vicinity of 0.5 meaning the density of 0s and 1s are comparable, then we proceed to the left-hand branch to examine the local correlation coefficient $\alpha_{0\to1}$. If $\alpha_{0\to1}$ is sufficiently small, then PRD is recommended for that column (label A); otherwise, constant should be selected to reduce bit-line switching activity (label B). If the input data set is sufficiently biased, i.e. if ρ is larger than some $0.5 + \delta$ for some δ, then we proceed to the right-hand branch. It should be noted that if the data set has a sufficiently high global correlation (label C) then its local correlation is also high by definition. An extreme example of this is a series of purely 0s or 1s. In this extreme case, the two modes perform equally well. However, constant mode is preferred due to its superior stability in the presence of

noise. For example, if one 0 value appears in the middle of a sequence of 1s, constant mode has equal or fewer discharging activities than PRD mode. Further simulations are required for data with no distinctive global or local correlation (label D). The parameters C, δ and τ are essential to the decision rule presented and are determined later in Section 4.2.2 via post-layout simulation.

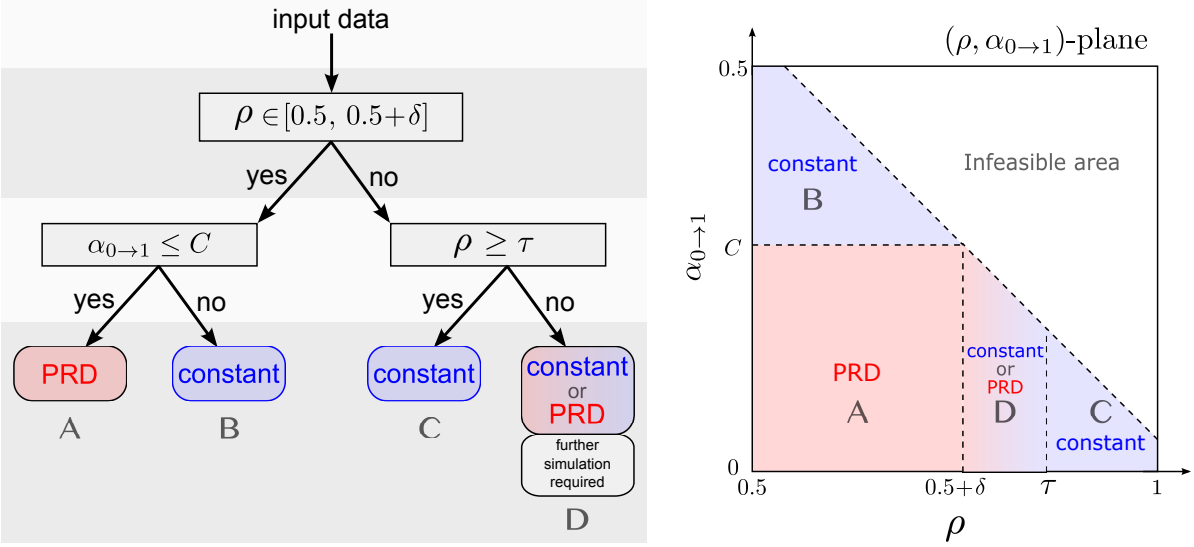

Figure 3-11: A decision tree (left) and an equivalent decision plane (right) illustrating the procedure by which prediction modes are configured for each SRAM column.

3.4.2 Proposed column circuit design

Figure 3-12 presents the column circuit for the reconfigurable CP SRAM. By adding a number of basic logic gates, e.g., multiplexers (MUX) and inverters, two prediction paths are introduced at the circuit-level. In particular, two inverters `I1` and `I2` are designed with skewed sizing to speed up the bit-line sensing. An And-Or-Invert (AOI) logic based Set/Reset (SR) latch takes the outputs of `I1` and `I2` along with the current prediction as inputs to resolve the read value. If one of the bit-lines stays high, meaning the prediction is correct, the SR latch holds the current values of `D` and `DB`; otherwise the SR latch inverts the values of `D` and `DB` reflecting that the read

value is complementary to the prediction. Once the latch outputs settle and one of the bit-line is pre-charged high according to the next prediction, an enable signal `dout_en` is asserted and a second latch, `I7`, latches `D` as the read output `dout`.

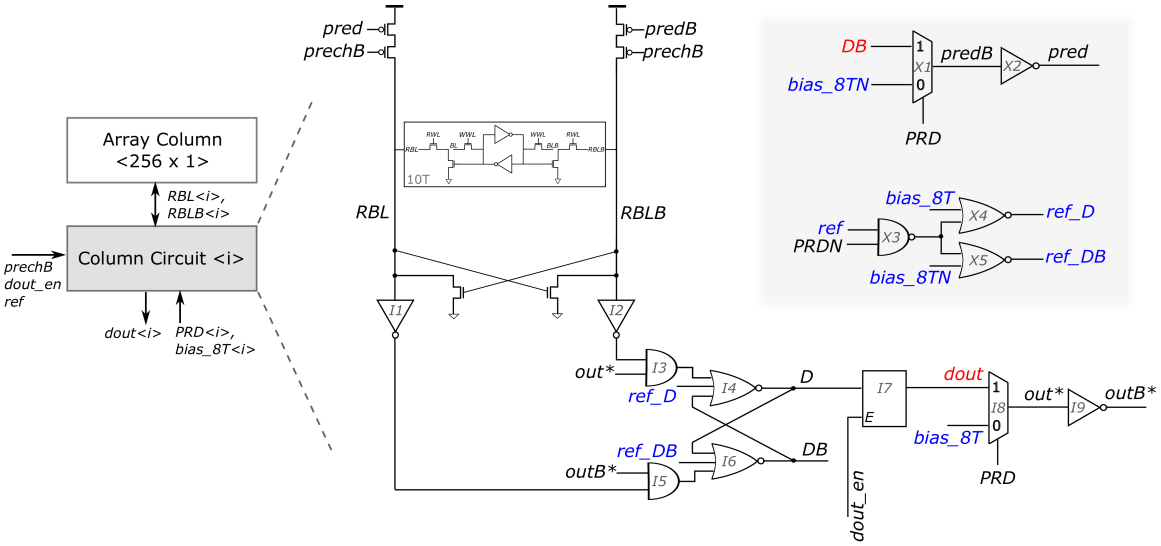

Figure 3-12: The proposed column circuit design which supports column-wise mode selection between PRD and constant prediction schemes. PRD signals are highlighted in red and constant signals are highlighted in blue.

While both prediction modes follow the above procedure for read operations, PRD and constant have separate prediction generation paths that are controlled by a mode selection signal `PRD` in MUXes `I8` and `X1`. When PRD mode is selected, i.e. `PRD` is set to high, the value of `DB` is selected to generate the new prediction and `dout` is passed to the SR latch as the prediction feedback `out*`. In constant mode, a DC bias control `bias_8T` indicating the majority bit of the column is sent to the prediction generation and feedback paths. Additionally, since constant mode invariably predicts 0 or 1 and pre-charges the same bit-line, it requires a reset in the SR latch every read cycle by using a control signal `ref`.

Chapter 4

Implementation of the reconfigurable CP SRAM

Consistent with the general framework outlined in Chapter 3, a 16kbit, 28nm CMOS test chip was developed using CP SRAM utilizing the 10T bit-cell illustrated in Fig. 3-2, the prediction-based conditional pre-charge scheme discussed in Section 3.2, and the reconfigurable column circuit depicted in Fig. 3-12. The primary focus of this chapter is to present the associated memory array and system architectures, discuss the implementation of the test chip, and provide a number of selected results from post-layout simulations. To begin, we discuss in Section 4.1 the specific architectures implemented at the array, macro system and chip levels as well as associated physical design considerations. In Section 4.2, post-layout simulation results are presented including functional verification of the system across process corners and power measurements for various operation modes and data sequences.

4.1 System architecture and physical design in 28nm FD-SOI CMOS

4.1.1 Memory array architecture

The SRAM array architecture is depicted in Fig. 4-1; the array is 16kbit in size and contains 256 rows and 64 columns of 10T memory cells along with row and column peripheral circuits. Specifically, memory cells in the same row share a Write Word Line (WWL) and a Read Word Line (RWL). During an active read/write cycle, one of these word lines is driven high through a word line driver depending on the specific operation performed. On the other hand, memory cells in the same column share a pair of write bit-lines (BL/BLB) and a pair of read bit-lines (RBL/RBLB). During a write operation, BLs and BLBs are driven to the desired bit values of input `DIN` by 64 write drivers. Each pair of RBL/RBLB is connected to pre-charge circuitry that consists of two PMOS transistors controlled by a global timing signal `prechB` and two other PMOS transistors controlled by a pair of column-specific prediction signals, `pred` and `predB`. As previously addressed in Section 3.4, each of these pairs of prediction signals is generated internally in a column circuit which also facilitates output sensing and configuration of prediction mode.

In the physical design of the memory array using a 28nm FD-SOI CMOS process, a custom 10T bit-cell is designed using logic rules specifically following the memory cell layout convention of two poly pitches as illustrated in Fig. 4-2. All gates within the peripheral blocks are also custom designed to ensure pitch matching, i.e. the height of each individual WL driver and the width of each column circuit and write driver is designed to match the height and width of the 10T bit-cell, respectively. Additionally, the metal routing design in the column circuit specifically minimizes the parasitic capacitance and hence delay and power consumption of critical timing signals such as the refresh signal `ref` and the output latch enable signal `dout_en` as shown in

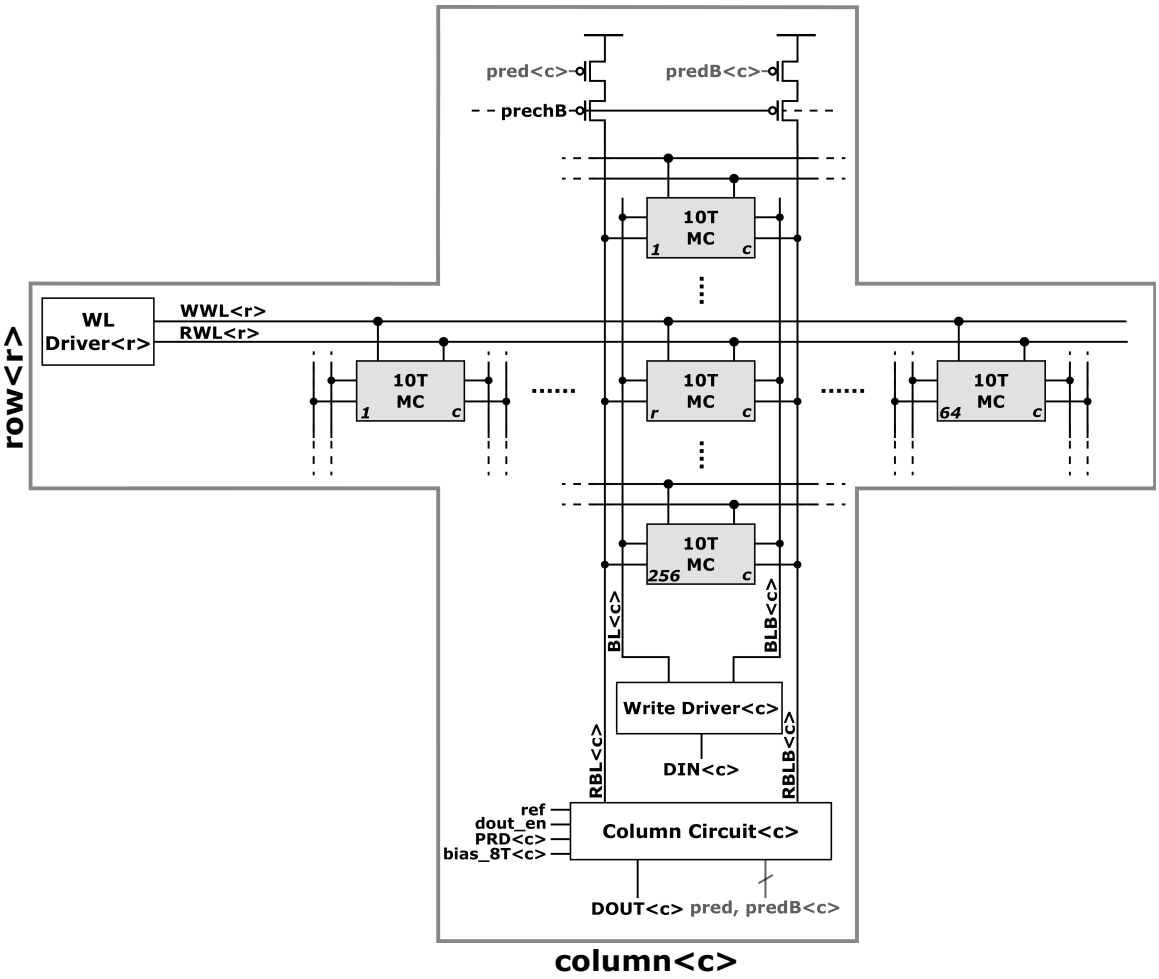

Figure 4-1: A diagram of the system architecture for the 16kbit memory array. The memory consists of 256×64 10T memory cells, row drivers for RWL and WWL, write drivers for BL/BLB, and a column circuit for RBL/RBLB variation sensing, prediction generation, and mode configuration.

Fig. 3-12. Write drivers are conservatively sized and not optimized for power since our focus is on reducing the disproportionate power consumption associated with read operations.

4.1.2 Macro architecture

The architecture of the macro system implemented on chip is depicted in Fig. 4-3. The specific column circuit design was discussed in Section 3.4. Equipped with separate

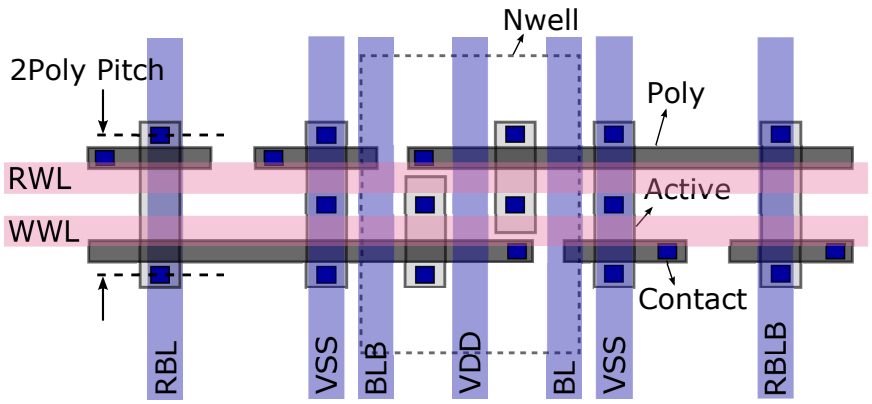

Figure 4-2: General structure of the custom 10T bit-cell. This diagram does not show the actual or any scaled version of the bit-cell layout. The approximate placements of NWELL, p and n active regions, polys, contacts, and METAL2/METAL3 routings are shown in the diagram.

read and write ports, the presented 10T array is capable of supporting simultaneous access by both operations. However, for the purpose of simplicity, read and write operations are designed to be mutually exclusive in this implementation meaning that only one or less of the two may occur during the same clock cycle. With this assumption in place, many peripheral blocks are shared between read and write operations which leads to savings in silicon area and reduced complexity of control.

An 8-to-256 two-stage decoder is implemented in order to decode the 8-bit input addresses for both read and write operations. Specifically, the first stage of the decoder contains two 4-to-16 sub-decoders which respectively take the most and least significant four bits of the address as their inputs. Outputs of these sub-decoders are hierarchically paired in the second stage in order to compute the 256 RWL/WWL enables for the array. The decoder is controlled by two enable signals labeled `rgate` and `wgate`. Depending on the specific operation, one of the enable signals is passed to the decoder in order to gate all input address bits and their complements in the first decoding stage so that the decoder would only start decoding when the gating signal is low. When the gating signal goes back to high near the end of an operation, the 256 decoder outputs, hence the 256 word lines, collectively go low. By using these

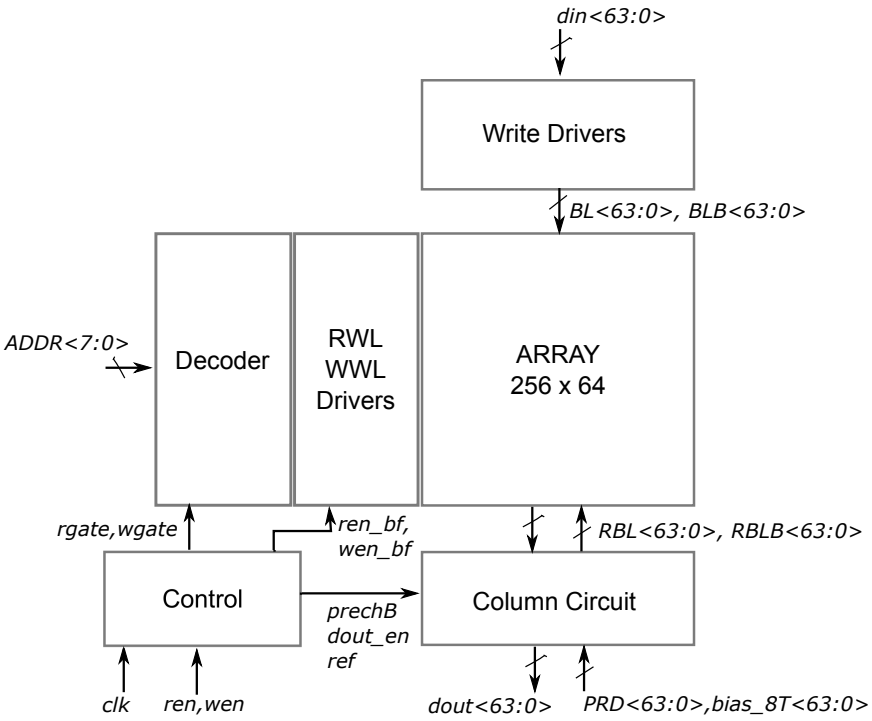

Figure 4-3: Architecture diagram of the system macro. The macro consists of the memory array, an 8-to-256 two-stage address decoder and a timing control unit.

gating signals, we ensure that all of the word lines are low during the transition from one clock cycle to the next, and thus preventing overwriting or reading at some wrong address while the new address is in the process of being decoded.

Another critical block of the macro system is the timing control unit. Referring again to the architecture diagram in Fig. 4-3, the control unit generates and passes to the array various timing signals including the aforementioned column circuit controls `prechB`, `ref`, and `dout_en` and decoder gating signals `rgate` and `wgate`. These timing signals are designed to have fixed-width pulses which are generated using delay elements along with the system clock (`clk`) and read/write enables (`ren`, `wen`). Subsequently, the system is robust in meeting the timing requirement of bit-line pre-charge and discharge activities even when the clock frequency is scaled.

Figure 4-4 depicts the resulting timing control waveforms for the sequence of operations: write, read, write, and read. Fixed-width pulses of `rgate` and `wgate`

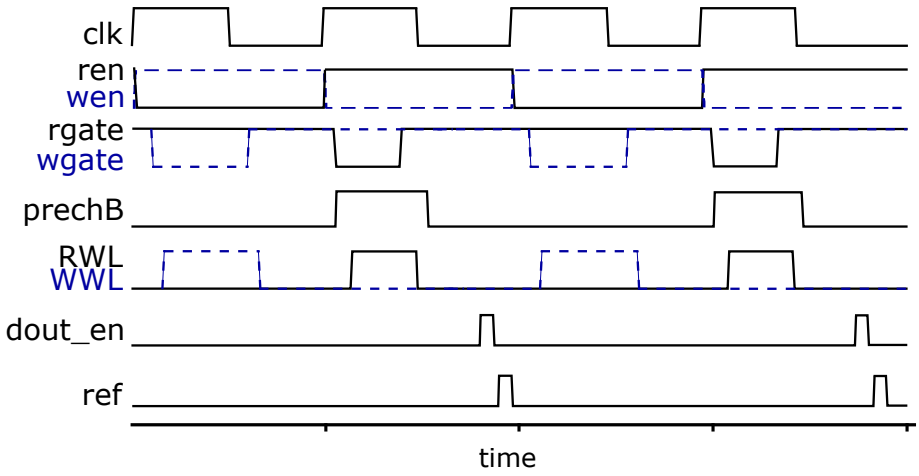

Figure 4-4: Timing control waveforms for the sequence of operations: write, read, write, and read. Write control waveforms including write enable `wen`, write decoder gating signal `wgate`, and write word line WWL are highlighted in blue.

are generated using delayed and inverted versions of `clk` and `ren` /`wen`. A falling edge of either `rgate` or `wgate` signals the beginning of address decoding for that specific operation. After a certain delay, the associated word line is pulled high as is indicated by the RWL/WWL waveforms. The pre-charge control signal `prechB` is kept low during write cycles indicating that the read bit-lines are constantly pre-charged to high. During a read operation, however, the edges of a delayed `rgate` create a fixed-width `prechB` pulse during which the read bit-lines are floated for bit value evaluation. Observe that the pulses on the RWL are strictly contained within the pulses of prechB, i.e. pre-charge is turned off during the entire time the RWL is high. This is purposely designed such that the effects of the pre-charge and evaluation activities on the read bit-lines are separated and the stored bits may be correctly evaluated through the column circuit without any disturbance from the pre-charge circuitry. The maximum operating frequency is considerably limited by the read bit-lines' pre-charge and discharge delays. Specifically, the time it takes to fully discharge a read bit-line through the stacked NMOS transistors determines the pulse width of the RWL while the fixed delay from the falling edge of the RWL to the rising

edge of the output latch enable `dout_en` is determined by the time it takes to fully pre-charge a read bit-line. For columns operating in constant mode, the pulses of `ref`, generated using the delayed and non-overlapping version of the pulses of `dout_en`, refresh the states of the corresponding AOI-based latches in the column circuit in preparation for the following read cycle.

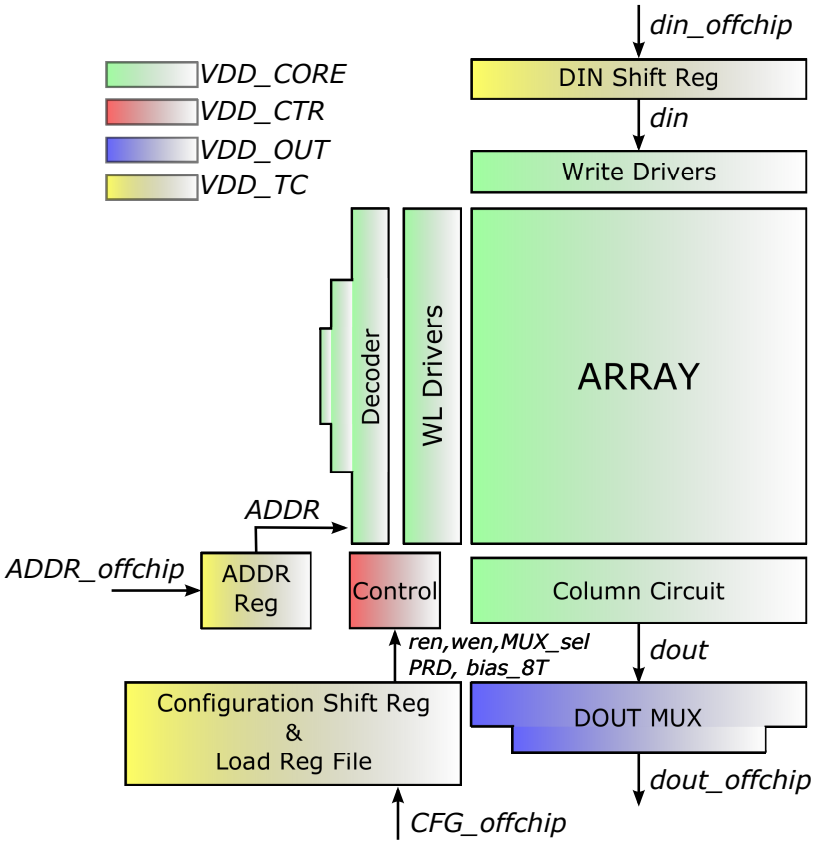

Figure 4-5: Architecture diagram of the test chip. Additional input/output and testing circuit blocks are added to the chip for functional completeness. Four supply voltages are indicated. Blocks supplied by the same V_{DD} are highlighted in the same color.

4.1.3 Test chip architecture

The macro system discussed above along with I/O and testing circuits constitute the complete test chip design. Figure 4-5 shows the architecture of the test chip. All address, data, timing, and configuration inputs to the system on chip are synchronized

to the clock's rising edge via I/O registers which include the DIN register file, the ADDR register file, and the configuration shift register. Limited by the number of available digital I/O pins, a 64-to-1 MUX is included for data output. Output data is readily synchronized through the output latches in the column circuit thus no additional registers are required. Observe that four different supply voltages are implemented on chip for the ease of testing and measuring power as is highlighted using four different colors in the architecture diagram. In particular, VDD_{CORE} (green) covers the memory array and its functional peripherals; VDD_{CTR} (red) supplies the timing generation unit and may be scaled for timing control; VDD_{TC} (yellow) supplies all of the input registers; and VDD_{OUT} (blue) is the power supply for the output MUX. Since no level shifters are inserted between these blocks, signals should only flow from blocks with greater or equal supply voltage than the destination blocks' supply voltage. In other words, the supply voltages satisfy the natural enumeration given by

$$VDD_{OUT} \leq VDD_{CORE} \leq VDD_{CTR} \leq VDD_{TC}. \tag{4.1}$$

Finally, Figure 4-6 shows the $1mm \times 1mm$ pad ring design and the completed full chip layout. The pad ring diagram on the left also shows the relative positions of the core system, the decoupling capacitors (DECAP), and the output level shifters (LS). Bonding pads around the external perimeter of the ring are highlighted in different colors to denote the different pads for: SoC power supplies (green), pad ring power supplies (red), system inputs (blue), and system outputs (yellow).

4.2 Post-layout results

Owing in part to today's aggressively scaled technologies such as 45nm and beyond, on-chip parasitics of digital circuits are often dominated by interconnect which does not scale as fast. Consistent with this remark, we observe a $1.5 - 1.8\times$ slower delay

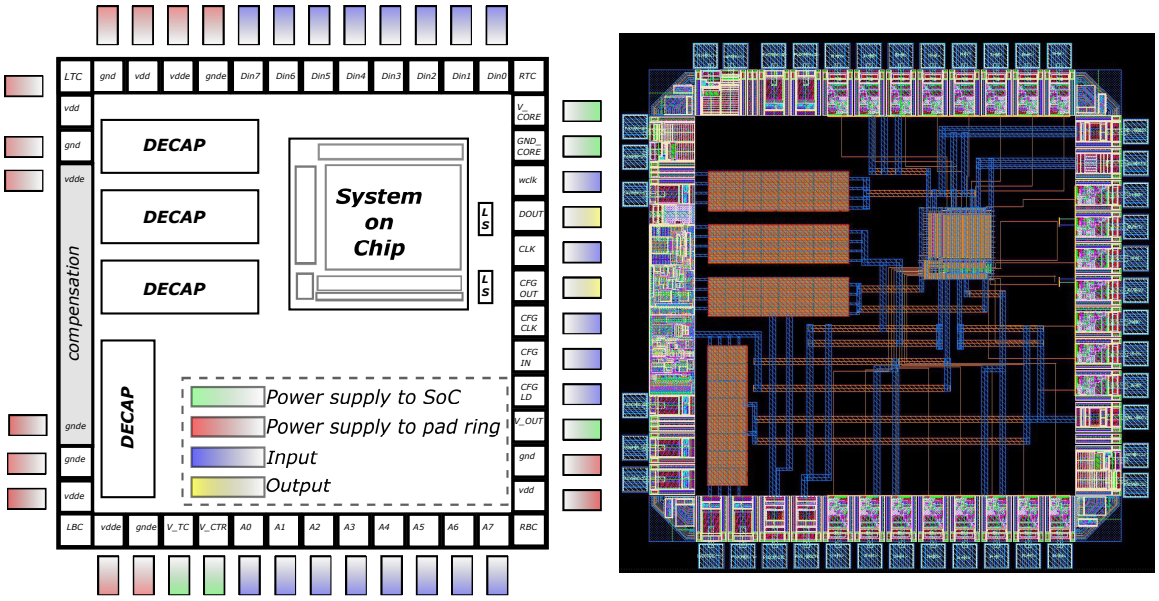

Figure 4-6: Left: pad ring design for the test chip. Bonding pads are highlighted in different colors to indicate the types of pad cells they are connected to. Right: completed layout of the test chip.

in post-layout simulation with extracted parasitic capacitances of the macro system as compared with the schematic simulation. Subsequently, delay elements and the supply voltage of the timing control unit are tuned through many design iterations to ensure proper functionality. In this section, we present our functional verification of the reconfigurable CP SRAM followed by a number of selected results from post-layout power simulations.

4.2.1 Functional verification

The macro system's functionality is tested with extracted parasitic capacitances across process corners with the supply voltage $V_{dd} = 0.65$V and clock frequency $f = 100$MHZ. The sequence of operations: (cycle i) writing a 1 in row 255, (cycle ii) writing a 0 in row 0, (cycle iii) reading the 1 from row 255, and (cycle iv) reading the 0 from row 0 is used to demonstrate the functionality of the reconfigurable CP SRAM operated in both PRD and constant modes. The waveforms corresponding to PRD mode at the

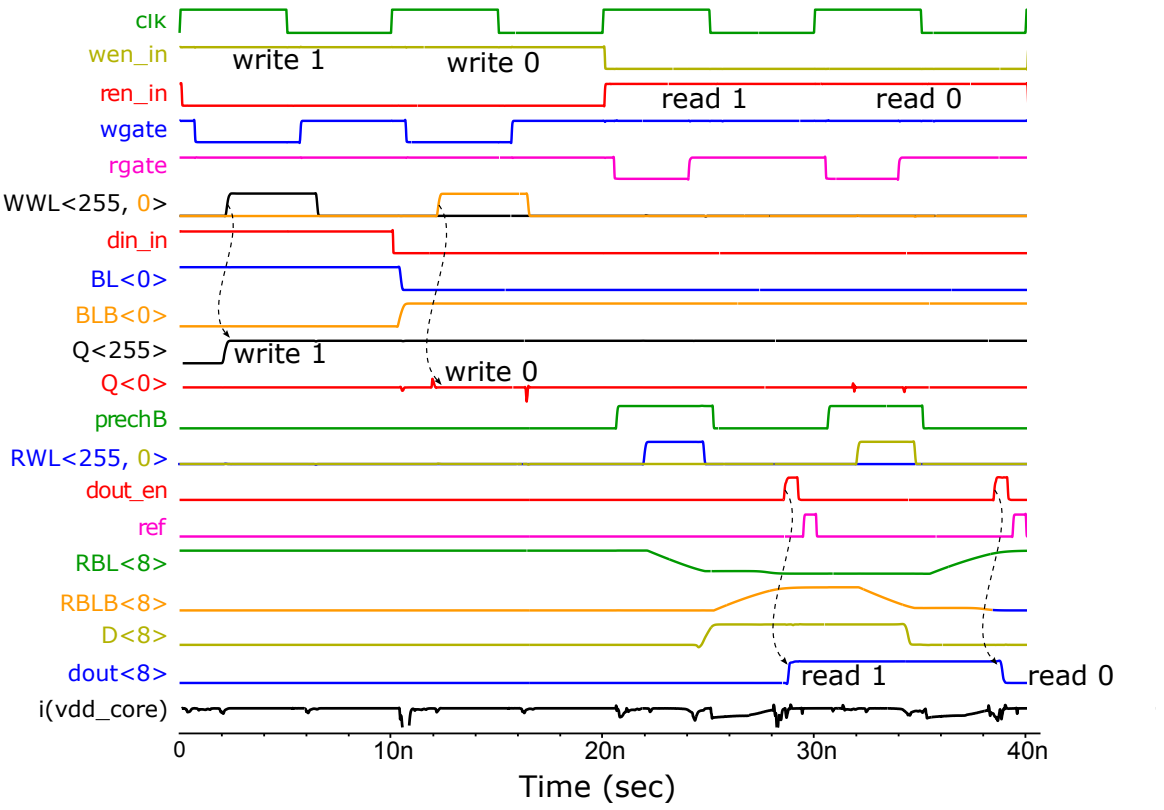

Figure 4-7: Simulation waveforms of the extracted reconfigurable CP SRAM operated in PRD mode. The sequence of operations performed are: (i) writing a 1 in row 255, (ii) writing a 0 in row 0, (iii) reading the 1 from row 255, and (iv) reading the 0 from row 0. The simulation is run with $V_{dd} = 0.65V$, $f = 100MHZ$, and at the SS corner.

slow-slow (SS) process corner are depicted in Fig. 4-7. In the first two clock cycles, input values 1 and 0 are successfully written to the internal nodes `Q` in row 255 and row 0, respectively. During the read operations in the third and fourth cycles, the read bit-lines RBL/RBLB are conditionally pre-charged based on current prediction values. The prediction value is initialized to 0 for the first read operation whereas the value 1 read in the first read cycle is used to update the prediction value for the second read cycle. For both read operations depicted, the prediction is readily seen to be incorrect. Subsequently, predicted read bit-lines discharge in both cycles, resulting in the observable two current surges in $i(vdd\_core)$. The simulation result for constant mode is illustrated in Fig. 4-8 where the prediction (bias) is set to be 0. After reading

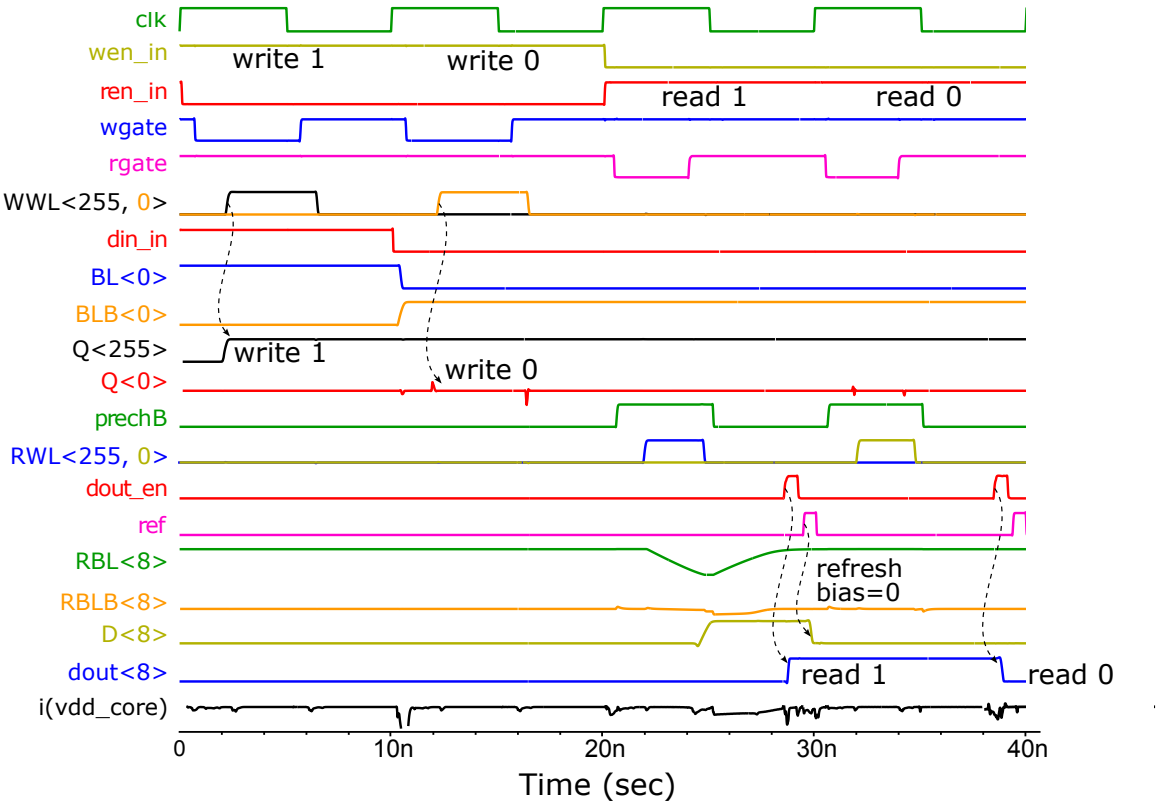

Figure 4-8: Simulation waveforms of the extracted reconfigurable CP SRAM operated in constant mode (prediction = 0). The sequence of operations performed are: (i) writing a 1 in row 255, (ii) writing a 0 in row 0, (iii) reading the 1 from row 255, and (iv) reading the 0 from row 0. The simulation is run with $V_{dd} = 0.65V$, $f = 100MHZ$, and at the SS corner.

a 1 in the first read cycle, the control signal `ref` resets the column circuit state `D` to be the bias value 0. RBL's discharge activity is saved in the second read cycle since the read value 0 matches the prediction value. As expected, $i(vdd\_core)$ has only one current surge in this specific scenario.

4.2.2 Power simulations

A power simulation model

The read power consumption of the extracted macro system is plotted in Fig. 4-9 using both PRD and constant prediction modes for all possible single-bit read values and

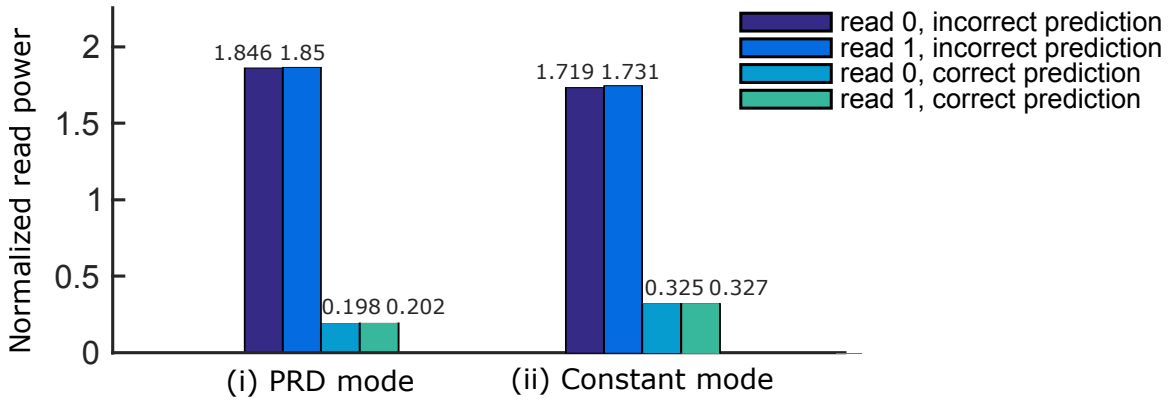

Figure 4-9: Simulated read power consumption of the reconfigurable CP SRAM operated in PRD and constant prediction modes for various scenarios. Specifically, four scenarios are simulated for each mode: reading 0 with an incorrect prediction, reading 1 with an incorrect prediction, reading 0 with a correct prediction, and reading 1 with a correct prediction.

prediction settings. Specifically, four scenarios are simulated for each mode: reading 0 with an incorrect prediction, reading 1 with an incorrect prediction, reading 0 with a correct prediction, and reading 1 with a correct prediction. Observe that in either mode, correctly predicted read operations incur much lower power than the same operations performed with incorrect predictions.

Due to the large size of the data sequences typical of our targeted applications, we simplify from hereon our power simulations by approximating the total power consumption of reading a prespecified long data sequence based solely upon the power consumption of the various read scenarios mentioned above.

Mode configuration

As previously discussed in Section 3.4, a decision tree may be used to optimally configure the presented SRAM. In order to identify the parameter values C, δ, and τ which are essential to the decision rule in Fig 3-11, we simulate the power consumption of the SRAM operated in PRD and constant mode for 25000 randomly generated data sequences each of length 52 bits. The simulation results are illustrated in Fig. 4-10.

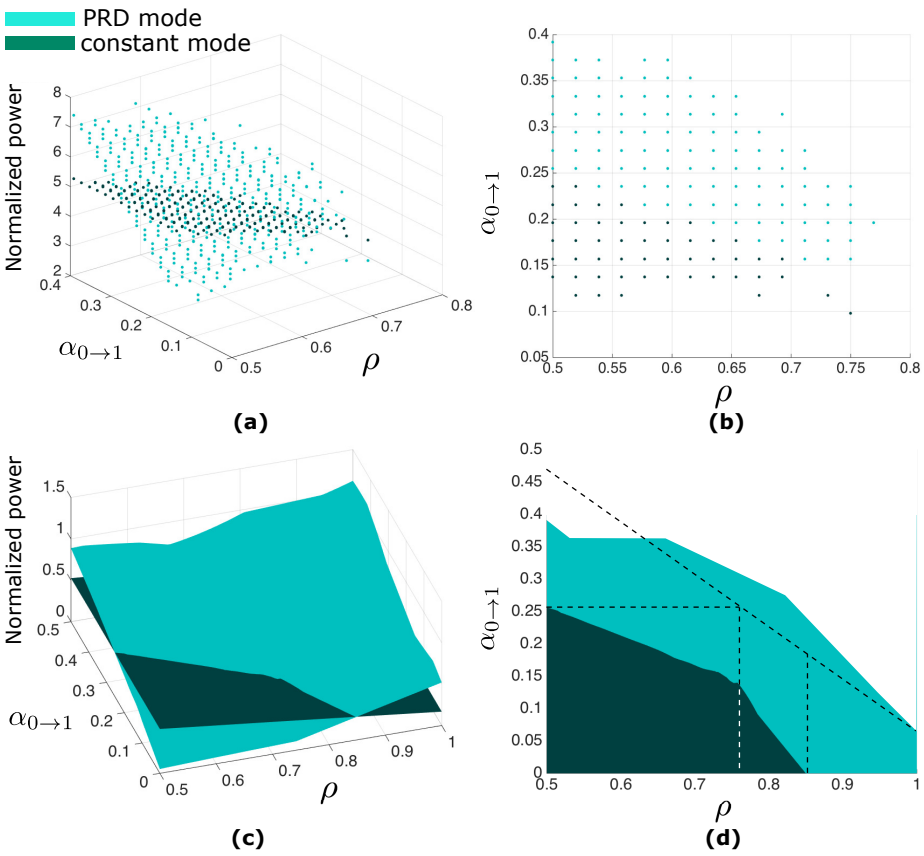

Figure 4-10: Mode selection analysis for randomly generated statistical measure pairs $(\rho, \alpha_{0\to1})$. Extrapolated data from (a) are shown in (c) and (d). The prediction mode corresponding to the higher power consumption is plotted in the plane views in (b) and (c).

Referring to this figure, subplot (a) shows the normalized read power consumption for each generated data sequence as a function of its statistical measures $(\rho, \alpha_{0\to1})$. Subplot (b) equivalently shows this result where the mode with the higher power consumption is plotted for each $(\rho, \alpha_{0\to1})$ pair in the $(\rho, \alpha_{0\to1})$-plane. To generalize our analysis, we extrapolate from the points in subplot (a) in order to predict the performance outcome over the entire $(\rho, \alpha_{0\to1})$-plane; the result is depicted in subplot (c). Analogous to the relationship between subplots (a) and (b), subplot (d) shows the projection of subplot (c) onto the $(\rho, \alpha_{0\to1})$-plane where the infeasible region has been excluded. This subplot is consistent with the $(\rho, \alpha_{0\to1})$-plane that was postulated in the discussion surrounding Fig. 3-11. In conclusion, we find that the decision

parameters C, δ, and τ are approximately given by

$$C \approx 0.25 \quad \delta \approx 0.26 \quad \tau \approx 0.85. \tag{4.2}$$

Chapter 5

Application-specific power results

The primary purpose of this chapter is to provide by means of example a method by which any SRAM consistent with the reconfigurable framework presented in this thesis may be properly initialized for a specific targeted application. In order to illustrate this general procedure, three example applications spanning digital signal processing, video coding and computer vision are surveyed in Section 5.2. Equipped with the power simulation model developed in Section 4.2.2, we compute the read power consumption of the reconfigurable CP SRAM for each of the targeted applications. Prior to this, in Section 5.1, we revisit the two key statistical measures introduced in Section 3.1 in order to facilitate the SRAM configuration stage for applications beyond those covered by the scope of this thesis.

5.1 Statistical data measures

The binary values processed and stored at the intermediary stages of an algorithm are often highly correlated. Chapter 3 illustrated that significant energy savings may be achieved for a given data set by toggling the SRAM configuration according to the data statistics. In particular, we introduced two key sufficient statistics, namely the global and local correlation measures ρ and $\alpha_{0\to1}$, from which a given pair $(\rho, \alpha_{0\to1})$

jointly corresponds to an optimal SRAM configuration.

Recall that the *global correlation* coefficient ρ of a prespecified data set was defined in Eq. 3.1 as the percentage of the majority bit, i.e. $\rho = \max\{p, 1-p\}$ where p is the percentage of 0s. It is generally possible to save power on the read bit-lines of an 8T SRAM when the bits stored in memory are mostly 1s. Figure 2-1 portrays that since 8T SRAMs' single-ended read ports require only one bit-line, the discharge path through `PG4` is open in the evaluation phase when the internal node `Q` stores a 1 and `QB` stores a 0. Thus, the pre-charge activity on the read bit-line is avoided in the following cycle; by symmetry it follows that comparable power savings may be achieved when the majority of the 8T bit-cells store 0s and the SRAM's read ports are connected to `Q`s. In order to obtain a higher global correlation coefficient, the bit-invert-coding method may be adapted using a majority logic conversion scheme wherein the total number of 1s stored in memory is increased by converting some words to their complements [8]. In contrast, the use of a majority logic conversion is unnecessary in the presented reconfigurable CP SRAM since energy savings may be achieved with a biased data set in either direction, i.e. one can select a desired read port via prediction setting in the constant mode.

Globally correlated data exists in a wide range of real-world applications where sparsity is present, e.g. matrices with a small number of non-zero entries or more generally where sufficiently small entries may be ignored. The use of sparse signal representation and companion algorithms can be found in tasks such as digital signal processing, compressive sensing and biomedical imaging [13]. The use of such methods and the associated theoretical developments are becoming increasingly practical as the demand for efficient processing of massive high-dimensional data such as audio, image, video and bioinformatics data increases. As an example, we studied a variant of the Fast Fourier Transform specific to sparse signals, i.e. the sparse Fast Fourier Transform (sFFT), and will discuss in Section 5.2.1 some of the challenges in the hardware

implementation of the algorithm as well as the memory power savings achievable by using the reconfigurable CP SRAM.

On the other hand, *local correlation* was defined as a measure of the bit-level changes of the signal values in a given data set. In part, it can be viewed as inversely proportional to the notion of the switching activity factor, an essential element of digital circuit dynamic power analysis and design optimization. The lower the switching activity factor, the higher the local correlation. We call attention to the fact that the complete definition of switching activity involves two parts: probability and toggle density. In this context, probability is the likelihood that a nodes voltage value is 1, whereas toggle density corresponds to the total number of $0 \rightarrow 1$ and $1 \rightarrow 0$ transitions seen at that node per unit clock cycle. For convenience, we focus only on the toggle density aspect of this definition since the probabilistic portion is covered by the global correlation coefficient. In addition, since power is drawn from the source only when the node of interest changes from 0 to 1 [16], the forthcoming discussion on switching activity will only account for the power-consuming transition of $0 \rightarrow 1$. Finally, conventional methods for computing switching activity are probabilistic in nature, e.g., under appropriate independence conditions $\alpha_{0\rightarrow1} = P_0 \cdot P_1$ where P_0 and P_1 are respectively the probability of a 0 and 1 occurring at a given node. However, for the purpose of data feature analysis in this thesis, we compute the switching activity of a prespecified data set according to Eq. 3.2 since we have access to the entire data set prior to SRAM operation.

It has been observed that high local correlation exists in data sets taken from multimedia applications such as video coding and image processing wherein the intensities of pixels situated in the same local region in an image or video frame are often similar in value. In order to explore the presence of such local correlation and other interesting data features, we focus our attention on two example applications: a motion estimation engine in a video coding system and a Support Vector Machine

(SVM) based object detection system.

We are specifically interested for the purpose of designing a data-dependent low power SRAM in applications whose computation involves data which exhibits local and/or global correlations. Only by understanding the inherent data features of such applications can we properly configure the SRAM in order to obtain higher energy efficiency in the data accesses of these systems. The two correlation measures ρ and $\alpha_{0\to1}$ will continue to play a key role in the discussions of the following section on the aforementioned targeted applications and their associated data characteristics.

5.2 Targeted applications and achieved power savings

As previously mentioned, three distinct applications are targeted in this work: computation of the sFFT, motion estimation in video encoding, and object detection for embedded vision applications. Previous hardware implementations of these systems, e.g. in [1, 32, 6], have proven to be significantly faster and more energy-efficient as compared with their software counterparts. Further analysis of each systems power breakdown indicates that SRAM access, more specifically read operation, accounts for a disproportionate fraction of total power consumption, and thus illustrates the opportunity for demonstrable power savings by equipping these systems with low-power SRAM modules.

In the following three sections, we consider one-by-one an example hardware implementation of each application and simulate the characteristic storage buffers in each system in order to profile their intrinsic data features. Then, we map the associated data sets typical of these buffers into the reconfigurable CP SRAM and compute the power consumption of reading them out using the power simulation model developed in Section 4.2.2. The individual columns in each of the targeted

buffers are configured to their lowest power mode as indicated by their statistical measures (ρ, $\alpha_{0\to1}$). The power consumption of the same CP SRAM operating entirely in PRD and the constant mode across all columns is also computed for comparison. As an additional reference, a conventional 8T SRAM is modeled using this CP SRAM with a data agnostic prediction scheme, i.e. the prediction value is taken to be 0 for each column.

5.2.1 Sparse FFT: collision resolution

The class of Fast Fourier Transform (FFT) algorithms is one of the most fundamental tools in signal processing, applied mathematics and computer science. Real-world applications such as wireless spectrum sensing, radar signal processing, and global positioning system (GPS) position locking [35] require complex computations involving the convolution of a signal with a long code. Such applications typically make use of large-sized FFTs in order to compute these convolutions while simultaneously meeting real-time constraints and thus result in high hardware costs for embedded or mobile systems with respect to silicon area and power consumption. In a subset of these applications, however, the signals being processed are often inherently sparse or approximately sparse in their frequency domain characterization, i.e. most of the Fourier coefficients computed by taking the FFT of the signal are either negligibly small or equal to zero, motivating a need for algorithms with exploit these properties to achieve lower complexity and higher implementation efficiency.

A novel algorithm, referred to as the sparse Fast Fourier Transform (sFFT) and first introduced in [10], has been shown to be computationally more attractive than traditional FFT algorithms for sufficiently sparse signal models. In essence, the sFFT computes the frequency representation of a sparse signal without sampling it at full bandwidth. It guarantees the sparse recovery of a signal with only a small subset of its samples, hence greatly reduces the cost associated with computation and intermediary

data storage.

The sFFT algorithm and the algorithm-inspired software implementation, the BigBand [11], follow the general procedure of bucketization, estimation and collision resolution. Bucketization is the process of partitioning an input signal into two signals via downsampling with co-prime rates and computing FFTs with co-prime sizes followed by placing the Fourier coefficients into the appropriate 'buckets.' Subsampling a signal in the time domain generally results in frequency domain aliasing. However, since the non-empty (non-zero or non-negligible) FFT coefficients of a sparse signal are concentrated at a limited number of frequencies, most buckets are either empty or have a single active frequency whereas only a few of them have collisions due to multiple active frequencies. The bucketization step is repeated for time-shifted versions of the input signal with a signal reconstruction process to follow. To reconstruct the full bandwidth, non-empty buckets are first passed to the estimation step, where the value and the position (frequency number f) of each active frequency are estimated. In particular, the value of a bucket with no collision is the value of the original spectrum at the frequency number of interest. The frequency number f is then calculated using $\Delta\phi$, the estimated phase change between the FFT coefficients of the input signal $x[n]$ and the time-shifted signal $x[n+\tau]$. This processing is summarized as:

$$x[n] \xrightarrow{N-\text{point FFT}} X(f) \tag{5.1}$$

$$x[n+\tau] \xrightarrow{N-\text{point FFT}} X(f)e^{j\omega\tau} \tag{5.2}$$

$$f \triangleq \frac{N\Delta\phi}{2\pi\tau}. \tag{5.3}$$

Equipped with the estimated frequency number f, collision detection is then performed for each bucket by comparing the values of the FFT coefficients corresponding to the input signal with and without a time shift. If the bucket contains a single active frequency, the time shift results in a linear phase change and preserves the magnitude.

Alternatively, buckets with multiple frequency collisions exhibit a change in both phase and magnitude. If no collision is detected, the estimated frequency value and position are assigned to the desired final spectrum and eliminated from further iterations. In the final stage of collision resolution, the above process is repeated between the two co-prime FFTs. It is known that with co-prime sampling rates, any two frequencies colliding in one bucketization are separate in the other bucketization. Once a single active frequency f is detected at bucket number b_1 in one bucketization, the corresponding bucket number b_2 in the other bucketization can be calculated by:

$$b_2 = f \bmod N_2 \tag{5.4}$$

where mod is the modulus operator, the remainder b_2 is the bucket number, and N_2 is the FFT size of the second bucketization. The sFFT algorithm is then able to locate and recover each sufficiently large frequency coefficient and eventually the full spectrum of the original input signal by iteratively finding and subtracting all buckets which, in the other bucketization, are collision free.

An existing hardware implementation

A VLSI implementation of the sFFT algorithm, as reported by Abari et al. in [1], demonstrated a 40× better energy-efficiency and an 88× speed-up in runtime as compared with traditional FFT algorithms implemented using an ASIC or Intel i7 processor when the signal is no more than 0.1% sparse in the frequency domain. As illustrated by the block diagram of the system in Fig. 5-1, two lengths of FFTs (2^{10}-point and 3^6-point) are selected for the bucketization step for a 0.75-million-point input signal. Specifically, an input signal subsampled by 3^6 in the time domain is passed to a 2^{10}-point FFT module and the same signal subsampled by 2^{10} is passed to a 3^6-point FFT module. Three FFT modules of each length are implemented to

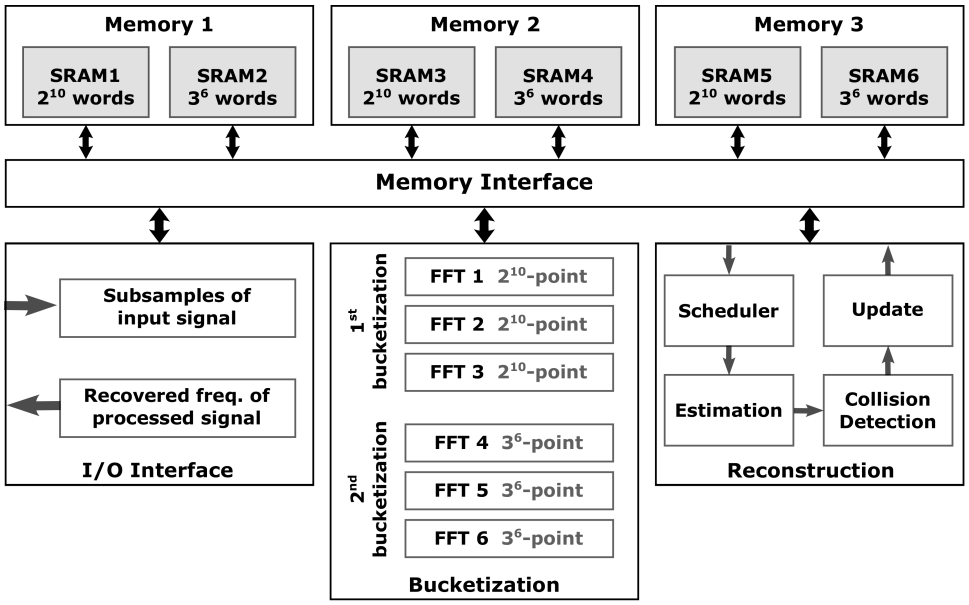

Figure 5-1: Architecture of the sFFT system [1]. Three memory modules are implemented to enable the efficient operation of a three-stage pipelined sFFT system.

compute the FFT coefficients of the input signal shifted in time by 0, 1 and 32 time units. The memory module consists of two SRAMs, each respectively consisting of 2^{10} and 3^6 rows, and is triplicated in order to enable the efficient operation of an associated three-stage pipeline that contains the I/O interface, bucketization, and reconstruction blocks.

Figure 5-2 shows the internal structure of the aforementioned two types of SRAMs used in this sFFT system. SRAM1 is designed to store the output coefficients of the three 2^{10}-point FFT modules in the first bucketization group. In particular, the FFTs of the subsampled original input signal are stored in the SRAM bank $b1$ whereas the FFTs of the subsampled and time-shifted versions of the input signal are stored in the banks $b1^{ts1}$ and $b1^{ts32}$ where the superscripts $ts1$ and $ts32$ are used to denote 'time-shifted by 1' and 'time-shifted by 32,' respectively. A similar bank division applies for SRAM2 which stores the output of the 3^6-point FFTs in the second bucketization group. Each of these SRAMs has a row size of 75 bits where 72 of them are used to store the three FFT coefficients, each represented by a 24-bit combined real and

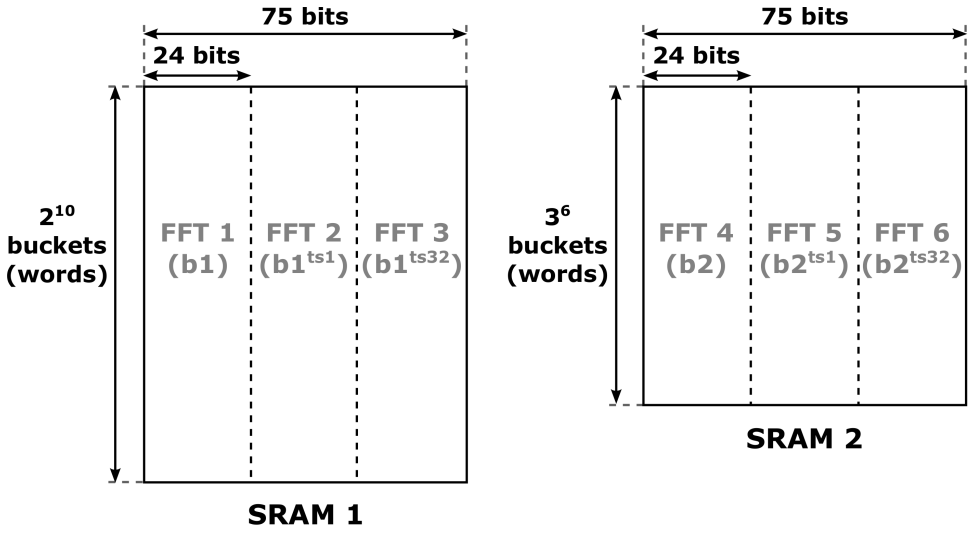

Figure 5-2: Architecture design of the two FFT coefficient SRAMs in the sFFT system. SRAM1 (left) stores the output of the three 2^{10}-point FFT modules in the first bucketization group; SRAM2 (right) stores the output of the three 3^6-point FFT modules in the second bucketization group.

imaginary value, and the extra 3 bits are the control bits for the system.

By using a three-stage pipeline and highly effective in-place FFT modules, this ASIC implementation of the sFFT algorithm is able to reduce its memory usage in the bucketization step and consequently lower the cost of hardware area and power consumption. However, it has been reported that over 50% of the total power consumption of the system is attributed to its on-chip SRAMs which are generated using generic memory compilers. Among the different operations that involve interaction with the memory block, the collision resolution step requires a significant amount of power for the almost *ad infinitum* retrieval of active frequency parameters from both types of SRAMs, thus motivating a need for more energy-efficient and application-specific memory designs.

An analysis of FFT coefficients in collision resolution

A key property exploited by the sFFT algorithm is the inherent sparsity of intermediary data. The majority of coefficients stored in the FFT SRAMs are either 0 valued or of

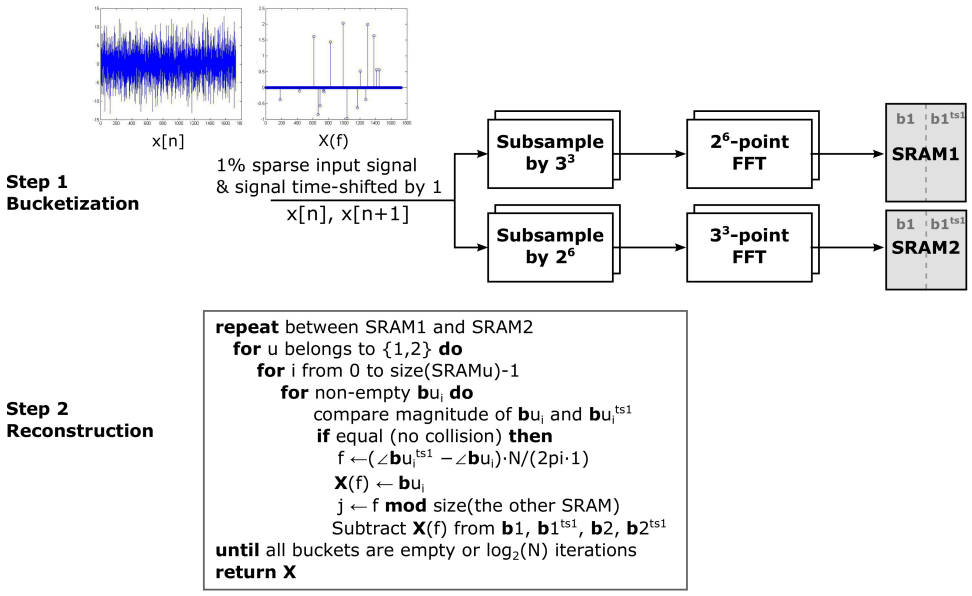

Figure 5-3: Architecture and pseudo code of a custom program that simulates data progression in the FFT coefficient SRAMs. Step 1 generates the SRAMs' input data and step 2 simulates the SRAMs' data progression during collision resolution.

negligible size. In order to visualize this sparsity, the data stored in each frequency bucket at every intermediate step of the sFFT algorithm is simulated using a custom program for a randomly generated sparse signal. Figure 5-3 shows the architecture and the pseudo code of the custom program, which is adapted and modified from the GHz-wide sensing and decoding algorithm in [11]. The simulation program follows the general steps of the sFFT algorithm described in the earlier section. For the purpose of an example, we simplify the hardware system in [1] to having only $2^6 \times 3^3$-sized input signals (up to 1% sparse) and hence 2^6 and 3^3-point FFT modules. The program, however, can be easily adapted for simulations of higher-point sFFTs with parameterized input signal length and sparsity level. In order to run the algorithm to completion and consistent with theory, the number of iterations of collision resolution is selected to be $\log_2 N$, where N is the length of the input signal to be analyzed.

Depicted in Fig. 5-4 is a typical progression of the data in the two FFT SRAMs highlighted in Fig. 5-3 over the course of the collision resolution procedure. The figure

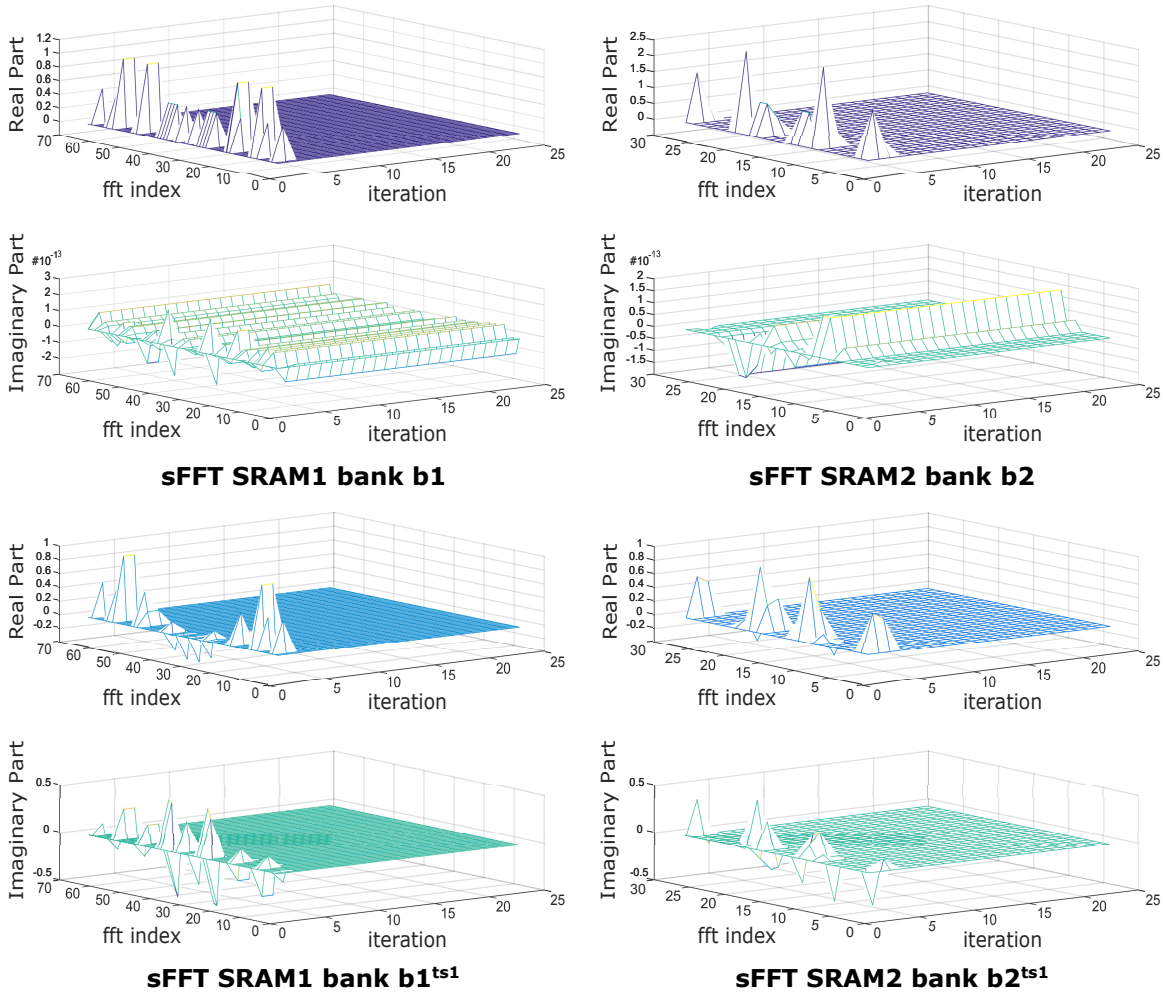

Figure 5-4: Storage data in different SRAM banks in the collision resolution stage ($2^6 * 3^3$ input size, 1% sparse in frequency). Real and imaginary parts of the FFT coefficients are plotted separately as they occupy different columns of the SRAM.

shows the sequence of values corresponding to the four SRAM banks $b1$ and $b1^{ts1}$ in SRAM1, and $b2$ and $b2^{ts1}$ in SRAM2. In particular, SRAM1 and SRAM2 stores the result of taking three 2^6-point FFTs and three 3^3-point FFTs respectively of the 1%-sparse, $2^6 \times 3^3$-length input signal after respective advances of 0 and 1 samples and appropriate downsampling. These four subplots illustrate that both SRAMs contain mostly 0s after 2-3 iterations of collision resolution. This observation suggests that using the constant mode of the presented reconfigurable CP SRAM which facilitates low-power read operations of biased values would be of merit. The sFFT application

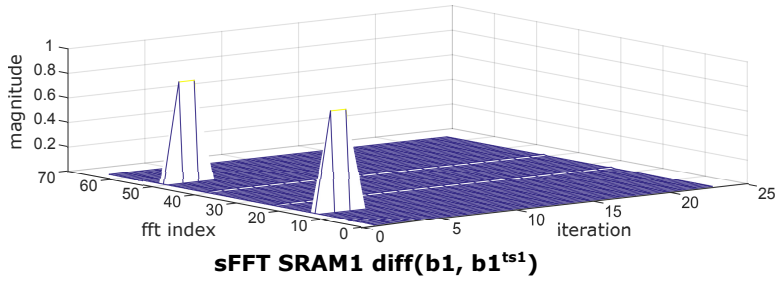

Figure 5-5: Magnitude differences between the FFT coefficients in SRAM banks $b1$ and $b1^{ts1}$ in a simulation of the sFFT collision resolution process.

maps well to the branch `C` in the decision tree in Fig. 3-11, where the embedded 'majority logic' of the constant mode further increases the number of 0 accesses from the FFT coefficient SRAMs.

Observe from the previously discussed simulations that low probability of collision as a result of a sufficiently sparse input signal ensures that the stored frequency information generated via systematic time-shifts of the original input are similar if not identical. Fig. 5-5 illustrates the difference in magnitude between the values in SRAM banks $b1$ and $b1^{ts1}$ throughout the collision resolution stage of a test simulation. Specifically, for a collision free bucket, the magnitude of the two coefficients are equal and their phases are linearly related. This observation provides a potential direction for future exploration of techniques that store the time-shifted FFTs such that data correlation within the SRAM columns is maximized. More 0 entries may be generated by storing only the differences between the FFT coefficients.

Power simulation result

In order to investigate column-wise data correlation and read power consumption within the FFT SRAMs, we convert the decimal representations of the FFT coefficients stored in SRAM1 and SRAM2 to their binary representations following the scheme in [1]. Specifically, each real/imaginary number is represented using 12 bits among which 4 bits are used for the fractional part.

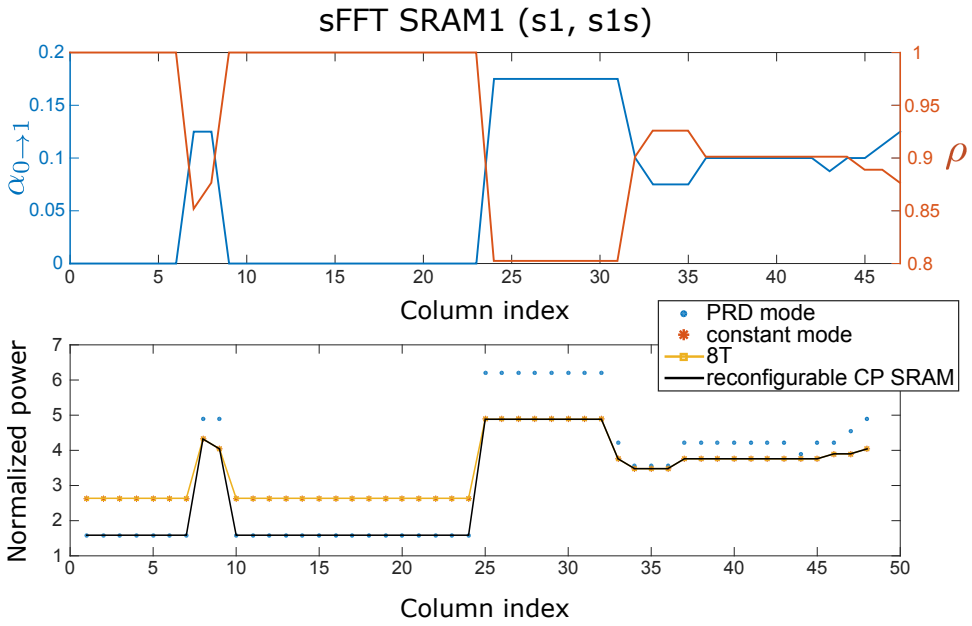

Figure 5-6: (Top) The statistical measures ρ and $\alpha_{0\to1}$, respectively computed in accordance with (3.1) and (3.2), as a function of each column of the data set associated with SRAM1 in an sFFT implementation. (Bottom) The normalized power for each column of the same data set for each of the listed memory models.

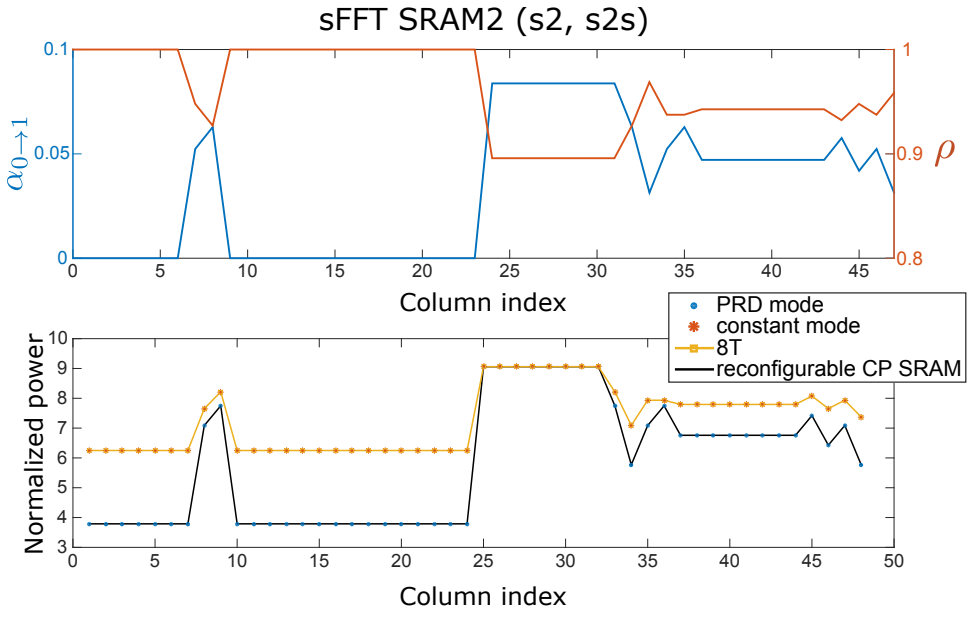

Figure 5-7: (Top) The statistical measures ρ and $\alpha_{0\to1}$, respectively computed in accordance with (3.1) and (3.2), as a function of each column of the data set associated with SRAM2 in an sFFT implementation. (Bottom) The normalized power for each column of the same data set for each of the listed memory models.

Following the power simulation procedure outlined at the beginning of Section 5.2, we select the preferred prediction mode for each column of the profiled SRAM1 and SRAM2 by first computing each column's statistical measures ρ and $\alpha_{0\to1}$. Figure 5-6 and 5-7 respectively show these statistical measures and the average read power consumption for each column of SRAM1 and SRAM2 implemented using the reconfigurable CP SRAM. The power consumption of the same CP SRAM operating entirely in PRD and the constant mode across all columns and the power consumption of a conventional 8T SRAM is also provided for comparison.

Still referring to the figures, 8T SRAM performs equally well as the CP SRAM restricted to the constant prediction mode. This is due to the fact that data stored in the sFFT SRAMs are mostly 0s and ρ is strictly larger than 0.8 for all columns in both SRAMs. The majority bit of each column is 0 and subsequently the constant mode predicts 0 for each column as does the modeled 8T SRAM. However, for some columns, PRD incurs lower power than the constant mode or 8T. Specifically in SRAM2 (Fig. 5-7), PRD results in lower power consumption than the constant mode for all columns except those for which $\alpha_{0\to1}$ is greater than ≈ 0.08. In SRAM1 (Fig. 5-6), more columns are configured to the constant mode due to their higher $\alpha_{0\to1}$. It should be noted that for columns with all 0 entries, i.e. $\rho = 1$ and $\alpha_{0\to1} = 0$, the simulated power consumed by PRD is lower than that of the constant mode. This gap is entirely due to the difference in the power consumed by the two modes even when they are reading a correctly predicted value. However, in the presence of noise, as is the case for columns 8-9 in both SRAMs, the power consumed by the PRD mode catches up with or even surpasses that of the constant mode. This observation is consistent with our discussion in Section 3.4.1 that constant mode is normally preferred for highly biased data sets due to its superior stability in the presence of noise. In summary, by effectively choosing the lowest power prediction mode for each memory column, reconfigurable CP SRAM achieves the lowest overall power consumption.

5.2.2 Video coding: motion estimation

Error tolerant and low latency distribution of high quality real-time multimedia content has experienced increasing demand in both rural and urban areas, continually pushing the boundaries of resource allocation protocols and technologies. In the age of big data, the transmission of such content in wireless networks is often the bottleneck. The use of a Video Encoder-Decoder (CODEC), a device or program capable of compressing and decompressing digital videos, is the most common and widely adopted remedy. One of the key challenges in the development of future multimedia systems lies in the design of efficient and quality-preserving compression algorithms and hardware architectures. Advanced Video Coding (AVC or H.264) and High Efficiency Video Coding (HEVC or H.265) are the two most recent video coding standards that target high compression capability. In particular, as compared with its predecessor H.264, HEVC targets a 50% gain in coding efficiency, equivalently a $40 - 50\%$ reduction in bit-rate, while maintaining the same visual quality. To achieve such goals, new tools are introduced such as the quad-tree data structure, a modular coding structure commonly used to decompose a video frame into separate regions to flexibly identify the optimal processing unit size. This flexibility in coding structure, however, comes at the cost of increased memory size and on-chip/off-chip bandwidth consequently leading to higher power consumption in hardware devices and systems. Various hardware-aware strategies have been proposed which allow for configurability and/or efficient resource sharing in hardware [38, 6, 27, 36]. Such efforts have typically causes degradation in the compression rate and quality achieved in practice, establishing a critical design trade-off between coding efficiency and hardware costs as is analyzed in detail in [28]. Subsequently, in order to lower energy costs without sacrificing efficiency at the algorithm and system architecture levels, we proceed with a focus on low power operations specifically at the circuit level.

We restrict our attention in this section to the design of video encoders whose

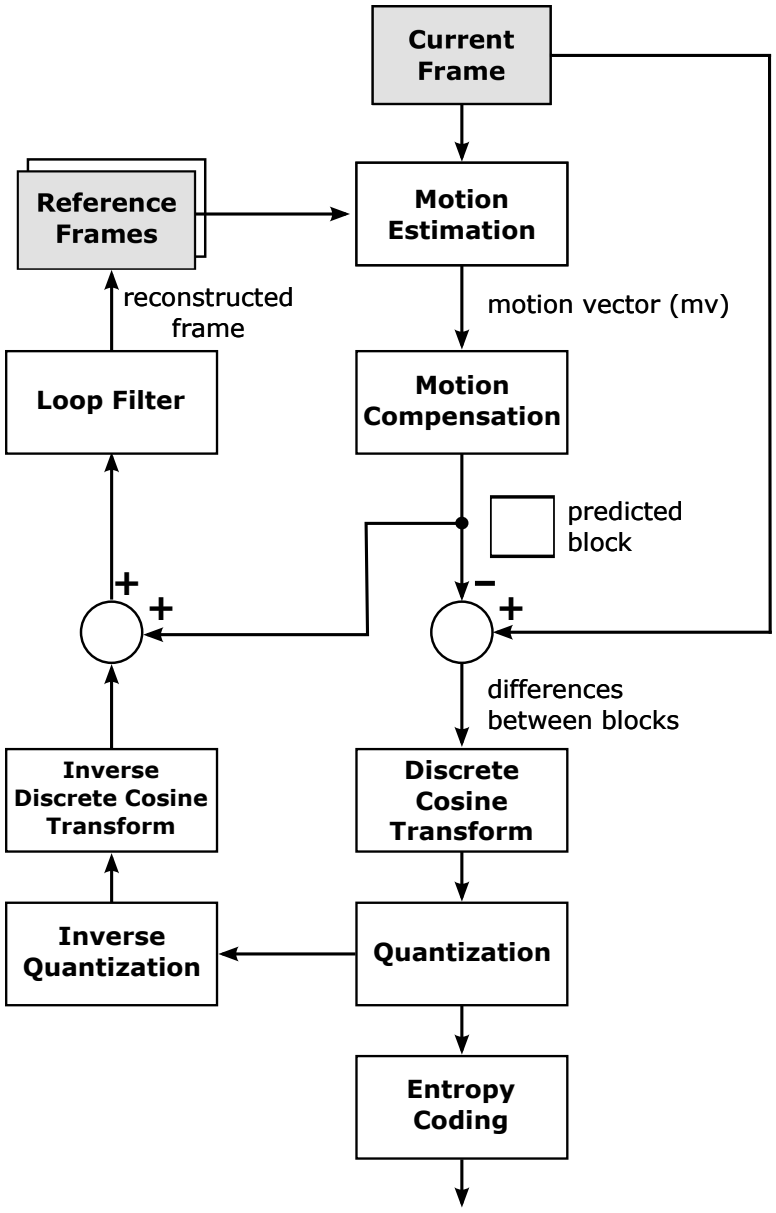

Figure 5-8: Block diagram of an example video encoding system with ME and MC engines. Motion estimation identifies motion vectors which are then used by motion compensation to retrieve predicted blocks in the reference frame. Processed differences between the predicted and the current blocks are transmitted to the decoder system which is not shown here. A current frame is also reconstructed at the encoder side which becomes the new reference.

primary responsibility in the overall CODEC is video compression. Figure 5-8 depicts a block diagram of an example video encoding system which makes use of inter frame prediction. In this encoder, Motion Estimation (ME) is one of the most critical tools

whose complexity directly affects that of the overall system since almost 90% of the total computation workload is attributed to the ME engine [19].

Motion estimation is a technique by which the movement of objects within a sequence of image frames is identified [14]. Given a previously encoded reference frame for comparison, the ME engine performs a motion search by computing, for each block in the current frame, the differences between the current frame and the reference for all blocks within a prespecified search range. After identifying the block of the reference frame with the most pixel-wise intensity resemblance (8-bit luma signals), the ME engine produces a motion vector which is later used in the motion compensation stage to locate the position of the predicted block. The computed differences between the predicted and the current blocks, i.e. the residual, are transformed, quantized and entropy coded before being transmitted to the decoder system. The residual in conjunction with the motion prediction vector are used to reconstruct the current frame at the decoder while the reconstructed current frame takes the role of the reference frame for the next iteration.

In tracking the movements of objects, the ME engine consecutively accesses on-chip reference pixel buffers in order to obtain the reference pixels necessary for computing Sums of Absolute Differences (SADs). As shown in Fig. 5-9, reference frames are generally stored using off-chip memory whereas a subset of the reference pixels are stored using an on-chip buffer that interacts with the ME engine directly. The duty cycle of conventional on-chip pixel buffers is disproportionately high and thus the power consumption is significant with respect to the overall system [6]. Moreover, the number of read accesses is greater than three times the number of write accesses to these buffers [27], providing motivation for novel memory designs which support low-power read operations.

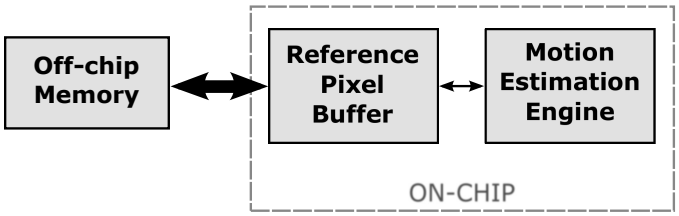

Figure 5-9: Off-chip memory and on-chip buffer for the ME engine. The engine consecutively accesses on-chip pixel buffer with a read to write ratio greater than three.

Existing frame buffer designs

Many energy and/or area-efficient memory designs are driven by the inherent statistical characteristics of video frame pixels in video processing systems, as previously addressed in Chapter 2. The existence of a large number of prior approaches is due in part to the high correlation between pixel intensities prevalent in pixels located in the same region of a frame. We refer to a digital group in what follows as the collection of a specific bit depth across all pixels in a given frame. Figure. 5-10 illustrates the analysis presented in [8] wherein the distribution of the number of 1s across different digital groups is plotted for two high-definition television (HDTV) test sequences, namely 'Market' and 'Church'. This work shows that adjacent pixels, specifically pixels used to represent the same foreground object or background setting, in an image are of similar intensity values and a stronger correlation exists in the more significant digital groups, i.e. the Most Significant Bits (MSBs). The majority logic SRAM in [8] leverages two key ideas: (i) the single-ended read nature of an 8T SRAM as previously addressed, and (ii) data-bit reordering and inverting in order to increase the number of 1s stored in memory. These efforts collectively save bit-line discharge activity and consequently the overall power consumed by the SRAM.

We take as a second and final example the local correlation analysis of pixel values in 16×16 processing blocks in a video frame, as was presented in [26]. Data extracted from the test sequences 'Traffic' and 'Basketball' is used in [26] to illustrate the

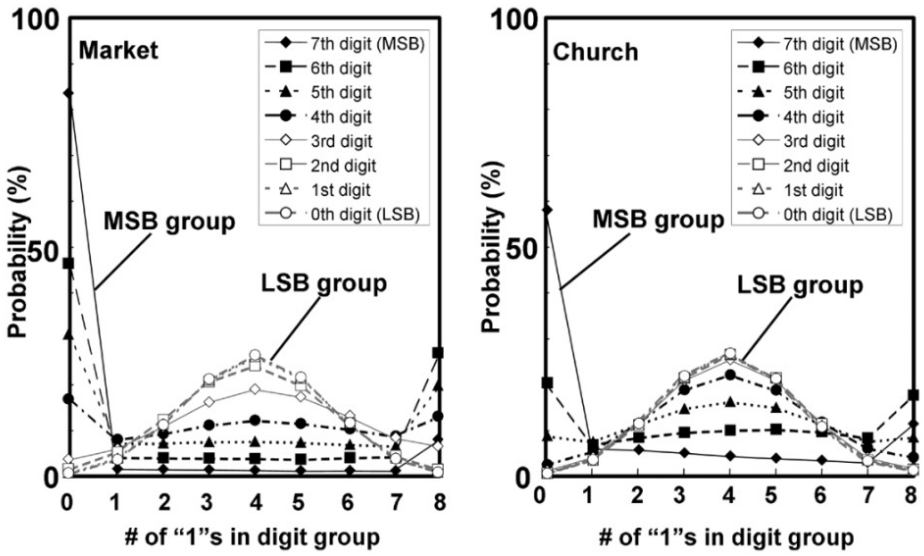

Figure 5-10: Statistical analysis of video frame pixels in H.264 HDTV test sequences 'Market' (left) and 'Church' (right). For each sequence, the probability of having different numbers of 1s is plotted for each digital group. Figure adapted from [8].

differences between individual pixels and the mean of the corresponding block using distributional techniques. It is concluded that for the two sequences 58% and 76% of the pixels, respectively, are within ± 3 bits of the corresponding block's mean. This implies that four to five MSBs out of the pixel's 8-bit depth are highly correlated with its neighbors' bit-values. Furthermore, the correlation among inter and intra frame pixels increases with increasing video resolution.

A statistical analysis of reference frame pixels

In order to gain further insight into the correlation between values stored in a typical reference pixel buffer, a custom program is developed to simulate and profile the effective data sequence that is accessed from the buffer during motion estimation. For the purpose of demonstration, we take a 1080×1920 image frame from the 'Park Scene' video sequence belonging to the JCT-VC common test conditions [3] as the simulation input. Figure 5-11 illustrates the data-to-memory mapping scheme selected

for this specific simulation. The entire reference frame is stored in an off-chip DRAM while the pixels in the current 192×192 search window are stored in an on-chip buffer implemented using an SRAM. The processing block size for current frames complies with the standard size of 64 pixels where the smallest computational unit, i.e. the sub-block, is selected to be 4×4. For each 4×4 sub-block read from the reference frame, we cycle through all sub-blocks of the same size in the associated processing block in the current frame for SAD computations. In order to meet a certain throughput requirement, the reference pixel SRAM is designed to support in each row 16 pixels, i.e. 128 bits in total. Depending on the compression rate of the transform and filter stages in Fig. 5-8, reconstructed reference frames may have different data correlation levels. We comment upfront that the input image frame used in this specific simulation is compressed with a high compression rate. With this setup, we simulate all read operations that are required to perform a complete motion search and then evaluate the local correlation between the values stored in each individual column of the SRAM.

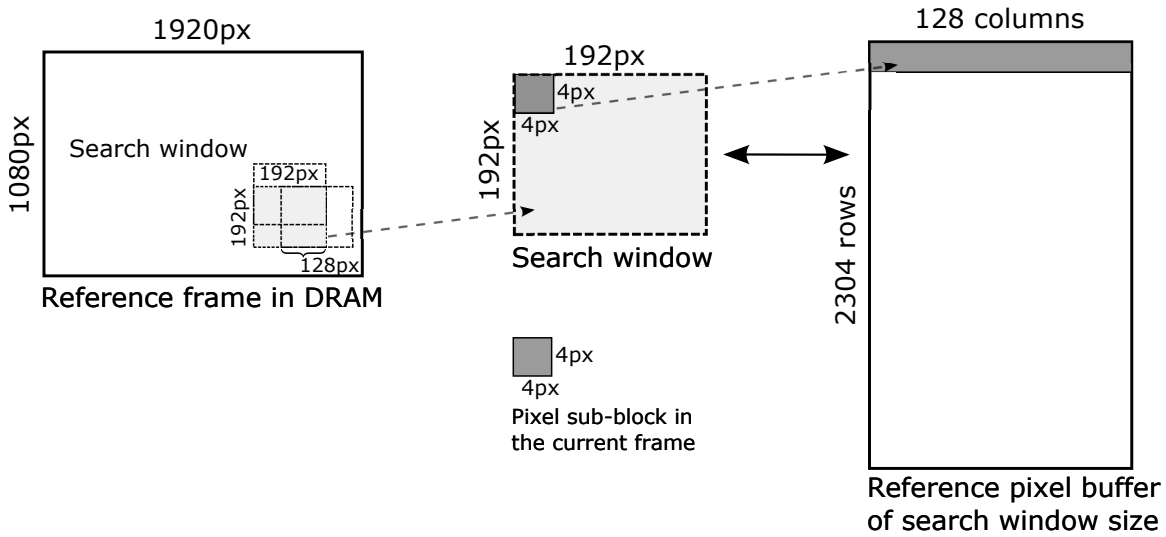

Figure 5-11: Diagram showing the data-to-memory mapping scheme. Pixels in each 192×192 search window within the reference frame are retrieved from an off-chip DRAM and stored in an on-chip pixel buffer implemented as a 2304×128 SRAM.

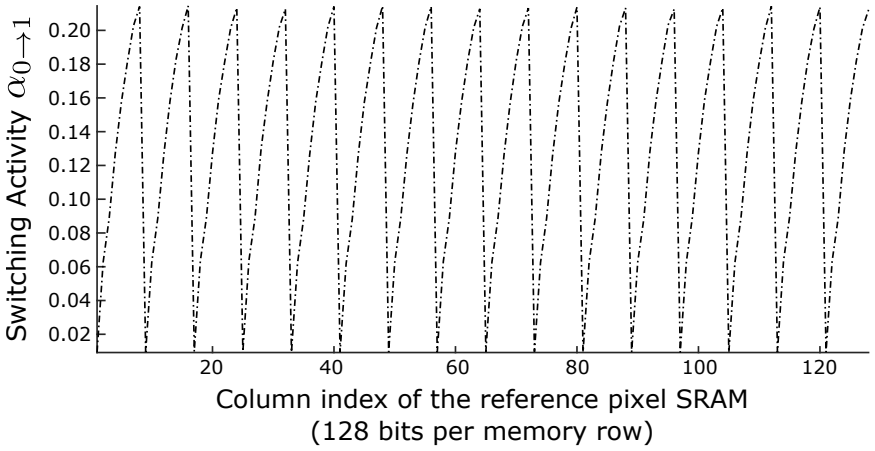

Figure 5-12: Switching activity in each column of the reference pixel SRAM. Pixels in a 1080×1920 highly compressed reference frame from the 'Park Scene' video sequence are stored in this SRAM which has sixteen 8-bit pixels per row, i.e. 128 bits per row. The MSBs of these pixels have much higher local correlation than the LSBs.

A plot of the simulation result is provided in Fig. 5-12. This local correlation analysis shows that observable switching activity patterns exist between the MSBs and the Least Significant Bits (LSBs) of the reference pixels. In particular, the activity factor $\alpha_{0\to1}$ of data stored in the column direction illustrates that the MSBs are more locally correlated than the LSBs, which are often corrupted by noise. This observation is consistent with the conclusions made in [8], and further implies that for each digital group, the switching activity is generally predictable.

Using the reconfigurable CP SRAM presented in this thesis, the read operations of the MSBs and the LSBs may be optimized separately, acknowledging their fundamentally different correlation statistics. This mode selection scheme is similar to the paths described by the decision tree branches `A` and `B` in Fig. 3-11. Additionally, equipped with the presented conditional pre-charge technique, predictions can be made and refreshed every read cycle with the PRD scheme, reflecting the bit-to-bit local correlation of the MSBs. For columns with high global correlation, i.e. heavily lopsided data such as the MSBs of the pixels in an image's background, PRD or constant mode with the correct bias setting can be utilized so that the power and area

overhead of an external inversion block and its controls can be saved.

We call attention to the fact that for video frame pixels, a corner case may occur where adjacent pixels are of similar intensity values but their 8-bit representations are very different. For instance, all bit values are changed in going from a pixel intensity of 127 (01111111) to a pixel intensity of 128 (10000000) [26]. These effects on local correlation can be mitigated by using a binary bit shifting technique [25]. For the purpose of simplicity in our system-level architecture design, we leave as future work the handling of these corner effects which may potentially further improve the bit-wise local correlation among video frame pixels.

Power simulation result

Analogous to the discussion in Section 5.2.1 surrounding the power simulation result of the sFFT coefficient SRAMs, we present the column-wise statistics and average read power consumption for each column in the ME reference pixel SRAM. As shown in Fig. 5-13 (top), the statistical measures do indeed follow the simulated patterns existing between the MSBs and the LSBs of the reference pixels. Specifically, low and high values of $\alpha_{0\rightarrow1}$ and ρ, respectively, are observed among the MSBs whereas the opposite case is observed for the LSBs. Referring to the bottom plot, similar patterns exist between the digital groups in terms of the power consumed by both PRD and the constant mode. For the video frame analyzed, the LSBs have a low average value of $\alpha_{0\rightarrow1}$ which is approximately 0.22 while the average value of ρ is close to its minimum value of 0.5. For this reason, the expected advantage of the constant mode among the LSBs does not manifest itself in this simulation result. This observation is consistent with the discussion surrounding the simulated decision plane in Fig. 4-10. We observe that the difference in the power consumed by the two modes gets smaller near the LSBs. A *mode slider* configuration scheme, i.e. using PRD for the columns storing the MSBs and the constant mode for the columns storing the LSBs, may be applicable for

video frames with more details. We conclude that as compared with the 8T SRAM, the reconfigurable CP SRAM achieves the minimum overall power consumption by correctly using PRD.

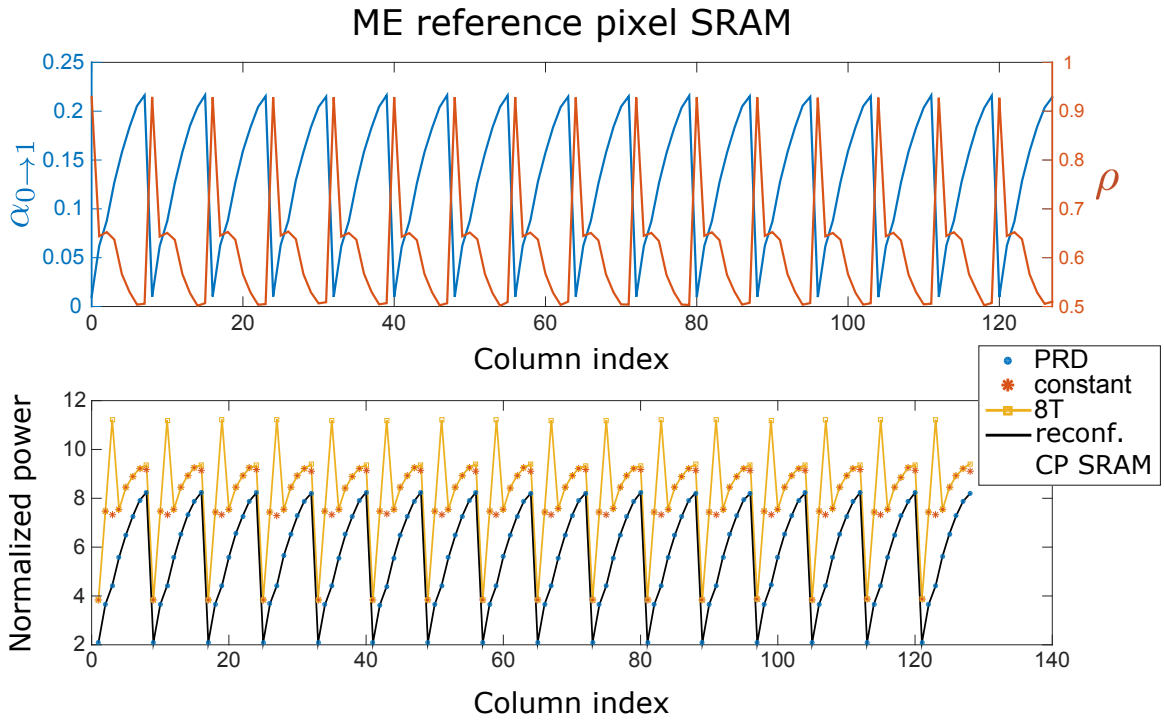

Figure 5-13: (Top) The statistical measures ρ and $\alpha_{0\to1}$, respectively computed in accordance with (3.1) and (3.2), as a function of each column of the data set associated with the ME reference pixel SRAM. (Bottom) The normalized power for each column of the same data set for each of the listed memory models.

5.2.3 Object detection: support vector machine classification

Object detection, the process of detecting particular instances of semantic objects in digital images or videos, has found application in a number of areas of computer vision including video surveillance and image database retrieval. As a straightforward example, consider the problem of detecting an object within a given test image or

video frame $\underline{a}$. The outcome is commonly generated as

$$\text{sign}(\underline{a}^T \underline{w}^\star - b^\star) = \begin{cases} 1 & \text{object detected} \\ -1 & \text{object not detected} \end{cases} \tag{5.5}$$

where $\underline{w}^\star$ and $b^\star$ denote respectively the weights and offset chosen for implementation. The specific values of these parameters are typically selected as the solution to the SVM training problem. More specifically, given a data set $\mathcal{D}$ where

$$\mathcal{D} = \left\{ (\underline{a}_i, y_i) : \underline{a}_i \text{ is the } i^{\text{th}} \text{ training image with associated binary label } y_i,\ i = 1, \ldots, M \right\}$$

and $y_i = 1$ if the particular object of interest is contained within test image $\underline{a}_i$ and $y_i = -1$ otherwise, the optimal weights $\underline{w}^\star$ and offset $b^\star$ are given by

$$\begin{aligned} (\underline{w}^\star, b^\star) = \arg\min_{\underline{w}, b} \quad & \|\underline{w}\|_2^2 \\ \text{s.t.} \quad & y_i \left(\underline{a}_i^T \underline{w} - b \right) \leq 1, \quad i = 1, \ldots, M. \end{aligned} \tag{5.6}$$

Extensions of Eq. 5.6 to multi-valued classification and therefore multi-valued object detection follow in a straightforward manner [2].

An existing hardware implementation

Modern advancements and hence imposed hardware challenges in the direction of embedded or mobile devices demand robust and energy-efficient hardware for object detection algorithms. Applications such as unmanned aircraft vehicles, portable electronics, and advanced driver assistance systems require real-time object detection with high frame rate, resolution, and energy-efficiency due to limited battery capacity and/or large heat dissipation [20]. In addition, such systems are expected to support multi-scale detection in order to facilitate detection of object of different sizes, e.g. a pedestrian walking from background to foreground.

Existing hardware implementations of object detection exploit a wide range of computational platforms such as CPU, GPU, FPGA and ASIC. Although relatively high throughput may be achieved using CPU or GPU-based systems, average power on the scale of hundreds of Watts is typically required for real-time processing with these platforms [22] and thus renders them as unsuitable choices for embedded applications. More energy-efficient high-definition object detection systems have been explored using FPGA or ASIC [9, 17, 33]. These implementations, nonetheless, require relatively large on-chip memory resources and therefore result in high hardware costs too.

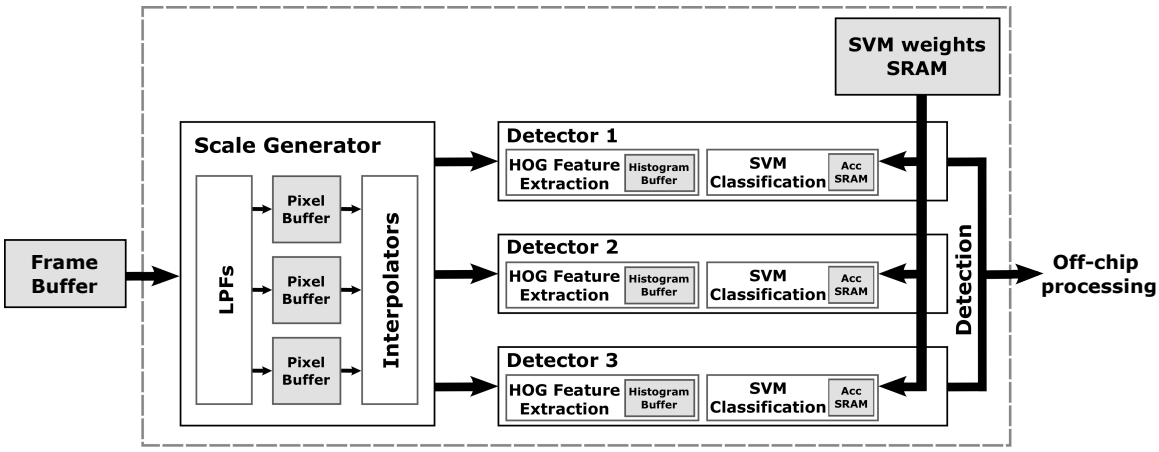

Figure 5-14: The core architecture of an example SVM based object detection system [32]. This system supports multi-scale detection and uses three detectors in parallel. Each detector performs HOG feature extraction and SVM classification for the inputted frames by accessing trained optimal SVM weights from an on-chip SRAM.

Alternative approaches in the literature address the challenges mentioned above by employing parallel detectors with balanced workload, on-the-fly SVM classification, and image pre-processing as depicted by the core architecture in Fig. 5-14 [32]. In this multi-scale object detection system, an original video frame image is first down-sampled to two octaves. Then, each octave including the original image generates three scales via bilinear interpolation, resulting in a 12-level image pyramid. These multiple scaled versions of the original frame are then fed into three parallel detectors,

each composed of a Histogram-of-Oriented-Gradient (HOG) feature extraction unit and an SVM classifier with optimal SVM weights trained off-line. A block diagram of the SVM classification unit is shown in Fig. 5-15. By using this on-the-fly multiply-and-accumulate approach, only the accumulated values of the partial dot products of HOG features and SVM weights are stored in the accumulation SRAM, leading to a 19× reduction in memory size.

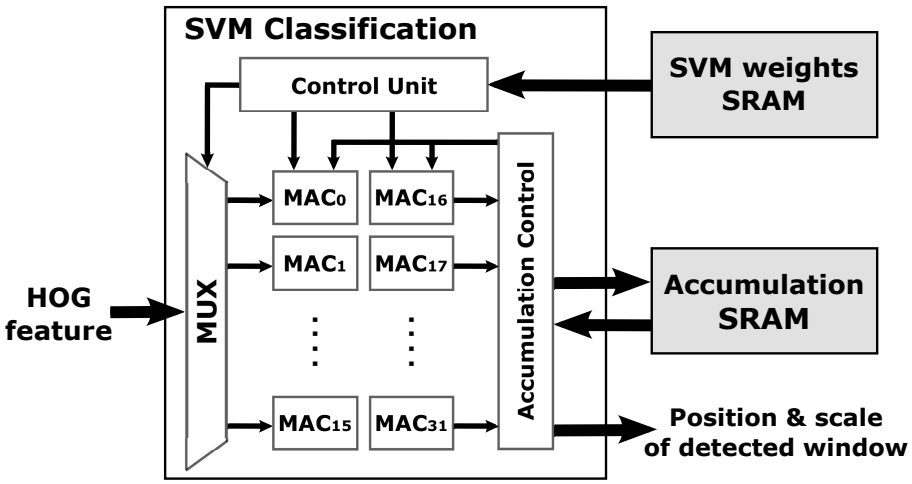

Figure 5-15: Architecture of the SVM classifier used in the object detection system in [32]. Thirty-two Multiply-And-Accumulation (MAC) modules and an accumulation SRAM are used to realize on-the-fly SVM classification.

Beyond the memory size savings achieved, the use of various on-chip memories, highlighted in grey in Fig. 5-14 for emphasis, imposes a number of system-level design challenges with the potential for improved energy efficiency. The SRAM used to store the SVM weights in the classification unit, for instance, is the current power bottleneck as a result of the large bandwidth requirement, i.e. the optimal SVM coefficients $\underline{w}^\star$ and $b^\star$ in Eq. 5.6 are accessed in both a frequent and consecutive manner from the SRAM for real-time classification processing. It has been reported that the power consumed by the SVM memory module is approximately 85% of that of a detector processing gradient magnitude images and comprises approximately 17% of the total

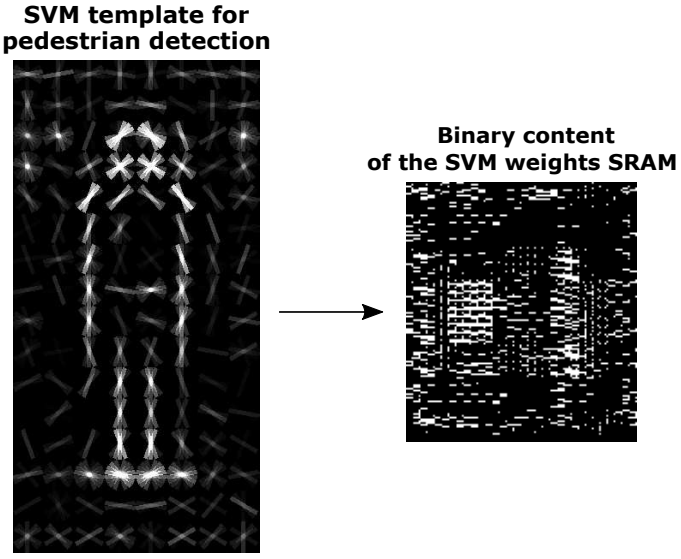

Figure 5-16: Visualization of SVM coefficients trained for pedestrian detection and their binary representations stored in the SVM weights SRAM.

power consumption of the system. We comment upfront that such read-only memories may be implemented using only register files. However, due to the area and density constraints of real-world systems, SRAM is typically selected as the storage medium for optimal SVM coefficients in a system.

A statistical analysis of optimal SVM weights

Analysis in [32] indicates that certain local correlation exists between the optimal SVM weight coefficients that are used to identify whether a specific feature and/or object is present. As an example, we analyze the SVM weights trained using a pedestrian image from the INRIA person data set, as illustrated in Fig. 5-16[1]. The particular weights corresponding to the general vicinity of image background are highly spatially correlated and therefore take similar values whereas the lighter coefficients corresponding to the object of interest form a pattern that generally follows the outline of a pedestrian. The binary representation of the weights stored in the SRAM is

[1]Data set and visualization program courtesy of the Energy-Efficient Multimedia Systems Group at MIT

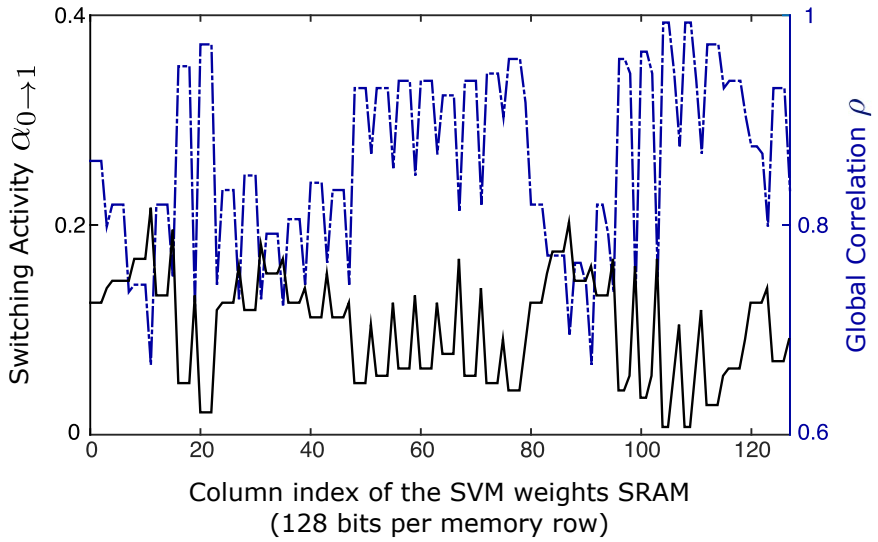

Figure 5-17: Column-wise correlation statistics of the SVM weights SRAM. Columns with low switching activity $\alpha_{0\to1}$ (black) tend to have high global correlation ρ (blue).

depicted on the right hand side of Fig. 5-16. Specifically in our simulation, we follow the original system's setup wherein the SVM classifier has a size equal to the number of HOG features in a detection window consisting of 16×8 cells, where each cell is a patch of 8×8 non-overlapping pixels. A total of $16 \times 8 \times 36$ signed 4-bit SVM weight coefficients are stored in a 18Kb SRAM with a row size of 128 bits.

Under the assumption that the SVM weights SRAM is fully written with the example optimal weights and is accessed for run-time classification row-after-row with incremental addresses, we compute the statistics ρ and $\alpha_{0\to1}$ for each column of the SRAM with the content depicted in Fig. 5-16. Plotted along the two vertical axes of the graph in Fig. 5-17 are respectively the percentage of the majority bit ρ and the switching activity $\alpha_{0\to1}$ in each column of the populated SRAM. Ticks on the horizontal axis are the column indices. It is evident from the figure that data stored in the SVM weights SRAM does not possess a straightforward switching activity pattern as compared with the reference pixels in ME. Additionally, the switching activity is low in columns where the percentage of the majority bit is high. The global and

local correlation measures are not statistically independent of one another, a proof of this argument is straightforward and can be shown for example by using the extreme counterexample where $\alpha_{0\to1}$ is taken to be zero. Therefore, in this specific example of optimal SVM weights, it is not immediately evident from the figure whether one correlation is higher than the other for certain columns. Other than a simple slider in the case of a ME reference frame buffer, further analysis is required for a fine-grained configuration for each column of the SVM weights SRAM. This configuration process may follow the paths described in the decision tree in Fig. 3-11.

In addition to the SVM weights, our analysis shows that local correlation also exists between the image pyramid pixels stored in the on-chip buffers in the scale generator. The argument in Section 5.2.2 that higher resolution of video frames or images implies a higher local correlation among both inter- and intra-frame pixels also applies here. In addition, gradient magnitude images contain even higher pixel-wise correlation as compared with images with unprocessed intensities [32]. Motivated in part by these observations, we remark that a potential direction for future extensions involving this application would involve equipping the scale generator with a low-power data-dependent SRAM model in order to potentially reduce the power consumed in reading the pixel values.

Power simulation result

Figure 5-18 shows the statistical measures and the average read power consumption for each column in the profiled SVM weights SRAM. Consistent with our analysis of the correlation statistics, the amount of power consumed by the columns in this SRAM displays a less traceable pattern as compared with that of the ME reference pixel SRAM. We observe that the prediction mode corresponding to the minimum power consumption rotates between PRD and the constant mode across all columns. The reconfigurable CP SRAM fully leverages the data statistics and achieves the

minimum total read power consumption by effectively connecting the lowest points in the bottom plot.

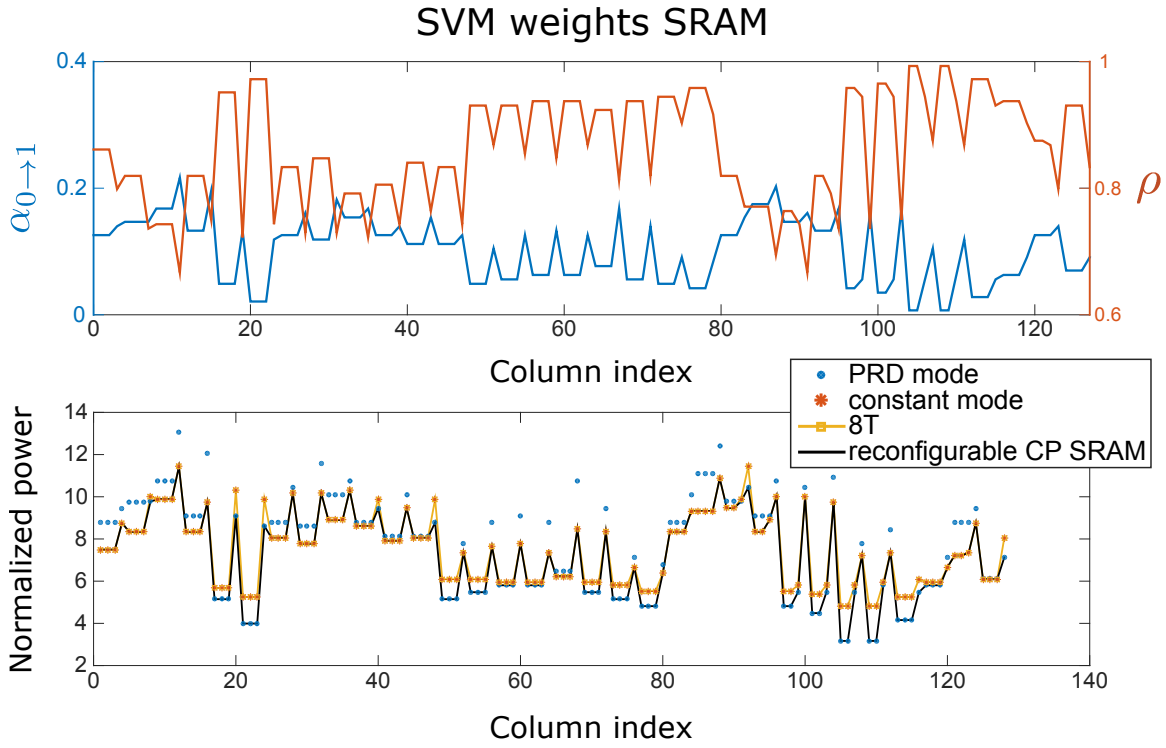

Figure 5-18: (Top) The statistical measures ρ and $\alpha_{0\to1}$, respectively computed in accordance with (3.1) and (3.2), as a function of each column of the data set associated with the SVM weights SRAM in an object detection system implementation. (Bottom) The normalized power for each column of the same data set for each of the listed memory models.

5.2.4 Summary of power results

As a summary, we present the normalized total (or average) read power aggregated across all columns for each application SRAM in Fig. 5-19. According to these results, the reconfigurable CP SRAM provides a respective 14%, 20%, 4%, and 31% reduction in read power as compared with the 8T SRAM for the SFFT coefficient SRAMs 1 and 2, the SVM weights SRAM, and the ME reference pixel SRAM. Also notice that with its columns dynamically configured, the reconfigurable CP SRAM has lower total power consumption than CP SRAMs equipped with only one of the prediction

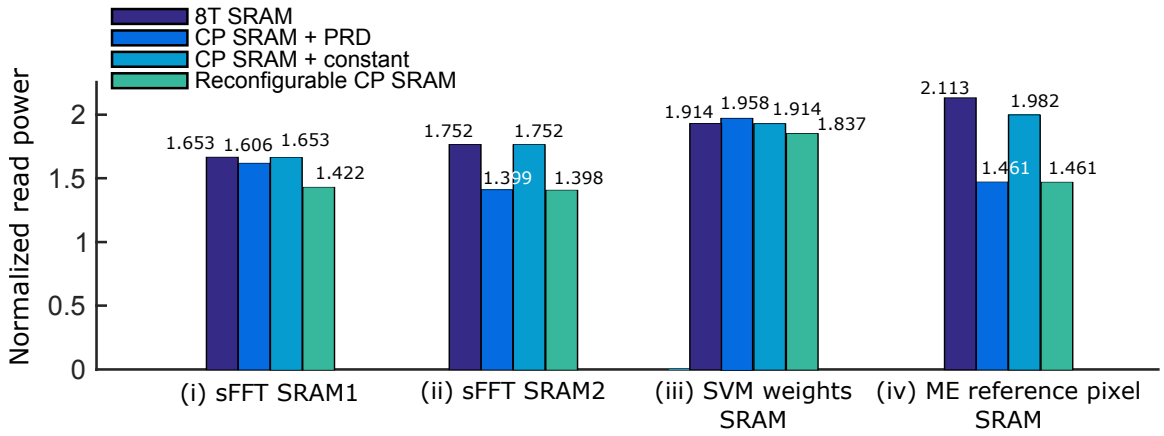

Figure 5-19: Simulated total read power aggregated across all columns of the data sets corresponding to the targeted applications for each of the memory models listed.

schemes. This observation is consistent with the results discussed in Section 3.3.3 obtained using the simulation program which in part motivated the reconfigurable memory design.

5.3 Further comments on data mapping

Statistical correlation present along a given direction of the data matrix stored in an SRAM depends critically upon the scheme by which application data is organized into the SRAM. For example, correlation between pixels in an ME reference frame buffer is directly affected by the prespecified search window and sub-block sizes as well as the dimensions of the SRAM used. Additionally, the specific sequence of read addresses affects how the data correlation is perceived by the SRAM's column circuit and consequently affects the prediction accuracy and total power consumption of the columns operating in PRD mode.

In the previous section, we discussed the method by which data is mapped into and accessed from the simulated SRAM(s) for each application independently. Each of these mapping schemes was adapted from an existing hardware implementation of the associated application and all applications were assumed to access data row after

row with incremental addresses. We reiterate here that a future direction is to explore alternative schemes in hopes of increasing the global and/or local correlation within the SRAM(s). For example, we postulated that storing the differences between the FFT coefficients in different SRAM banks of the sFFT system would result in an increase in the total number of 0 entries. These types of data organization and manipulation schemes, however, may require a modification of the application's algorithm and associated system architecture which may introduce considerable overhead in hardware and degrade overall performance. For this reason, data mapping should be judiciously designed for any application targeted. The details of this process for the applications considered in this thesis is outside the scope of our presentation.

Chapter 6

Conclusions

The primary focus of this thesis is the design of energy-efficient, data-dependent SRAMs equipped with reconfigurable prediction schemes and the methods by which application data statistics may be utilized to identify optimal configurations. On-chip memories, typically implemented using SRAMs due to their merit of high density, are known to consume a disproportionate amount of power in today's energy-constrained integrated systems. A commonly discussed approach to reducing the power consumed by these memories consists in lowering the supply voltage thereby inevitably introducing stability issues for the SRAMs and hence requiring additional circuit assist techniques to ensure proper functionality. This thesis addresses low power SRAM design in application specific settings by proposing a reconfigurable data-dependent memory framework that can be used to reduce the overall bit-line switching activity in the SRAM. As a demonstration of this framework, a conditional pre-charge 10T bit-cell paired with a reconfigurable dual prediction mode architecture was designed. A 16kbit SRAM test chip incorporating this innovation was developed using a 28nm FD-SOI CMOS technology. To optimally configure the presented SRAM for an arbitrary application, this thesis additionally provides tools for statistical data analysis and applied them to three example applications spanning digital signal processing, video

coding, and computer vision. In order to facilitate the extension of this framework to new applications, a systematic approach to configuring the prediction mode for each SRAM column was developed.

In the remainder of this chapter, we summarize the key ideas and contributions presented in this thesis and discuss a number of potential directions for future work.

6.1 Reconfigurable memory framework

In Chapter 3, a reconfigurable memory framework was presented wherein the communication between software and memory hardware via a reconfigurable prediction generation unit is emphasized. In this framework, application specific information including data statistics are obtained using software in order to dynamically configure memory hardware into its most energy-efficient mode, e.g. the mode which generates the highest number of correct data predictions resulting in reduced overall bit-line switching activity. In order to provide quantitative notions for the types of data statistics we are interested in, the global correlation measure ρ as well as the local correlation measure $\alpha_{0\to1}$ were defined. Complete definitions and interpretations of ρ and $\alpha_{0\to1}$ are provided in the discussion surrounding Eqs. 3.1 and 3.2 in Section 3.1, respectively.

Continuing our discussion of this framework, a thorough analysis of the data-dependent features of various memory designs was conducted via a custom dynamic read power simulation program (Section 3.3). The utility of this analysis was in evaluating the tradeoffs amongst these data-dependent memory models and then identifying the best suited one(s) with respect to the presented framework. Specifically, the simulation program computes the total power consumption of each SRAM column's read bit-lines parameterized using the bit-line capacitances and bit-line voltage swing for any SRAM data input provided in one of three ways: specified by the user,

synthesized using correlation parameters ρ and $\alpha_{0\to1}$, or profiled from an application.

In order to illustrate the procedure of organizing application data into the proposed memory and configuring its prediction modes, a decision tree and the equivalent decision $(\rho, \alpha_{0\to1})$-plane are provided in Section 3.4.1. Values of the statistical parameters essential to the presented decision rule were determined in Section 4.2.2 via post-layout simulation. As discussed in Chapter 5, this thesis also profiled data features for three example applications: collision resolution in the sFFT algorithm, SVM classification in object detection, and motion estimation in video coding. Results of this statistical analysis were used to configure the prediction mode for each column of the associated SRAMs independently.

6.2 Reconfigurable CP SRAM design in 28nm FD-SOI CMOS

An example SRAM adhering to the proposed reconfigurable framework was the central focus of Chapter 3 and is referred to as the reconfigurable CP SRAM. Specifically, a 10T memory bit-cell was developed for use with a novel conditional pre-charge scheme that allows for power to be saved at the bit-lines whenever data is correctly predicted. This memory array design significantly reduces the power overhead associated with prediction switching in [26] and was the primary motivating factor for the development of the PRD prediction scheme wherein the previous value read is used to generate the new prediction value. Using our read power simulation program, PRD was proven to save total bit-line power for data sets with various statistical features as compared with arithmetic averaging using variable length window sizes. We also evaluated the performance capabilities of the proposed CP SRAM using PRD against three existing data-dependent memory models. A key conclusion from this evaluation is that combined CP SRAM and conventional 8T SRAM results in the most power savings

when the data is globally and/or locally correlated. This observation was incorporated into our final design of a reconfigurable memory where column-wise prediction mode configuration is made readily available. A hierarchical depiction of these proposed contributions is illustrated by Fig. 6-1.

In Section 3.4.2, we presented a circuit-level implementation of the reconfigurable CP SRAM. In particular, a compact column circuit endorsing reconfigurability was developed in order to facilitate fast and robust bit-line sensing, prediction generation, and prediction mode selection for each memory column independently. In Chapter 4, the implementation of a 16kbit reconfigurable CP SRAM test chip in a 28nm FD-SOI CMOS process is discussed. Architecture designs at several hierarchical levels are presented as well as the associated physical design considerations. Finally, we provided a number of selected post-layout simulation results which include the functional verification of both operating modes of the SRAM and the simulated power consumption of all possible single-bit read values and prediction settings. Using the power model discussed in Section 4.2.2, we approximated the average read power consumption of our proposed reconfigurable CP SRAM for the previously mentioned applications and compared the result against three other SRAM models. The reconfigurable CP SRAM was found to provide 14%, 20%, 4%, and 31% reductions in read power as compared with a modeled conventional 8T SRAM for the SFFT coefficient SRAMs 1 and 2, the SVM weights SRAM, and the ME reference pixel SRAM, respectively.

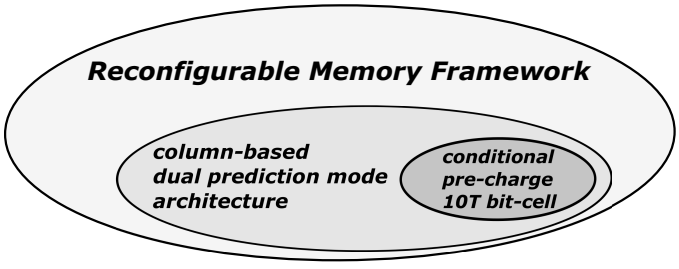

Figure 6-1: Hierarchy of the main thesis contributions.

6.3 Future directions

We now state a number of potentially fruitful directions for continued research. A more rigorous analysis of each of the data-dependent SRAM models addressed in Section 3.3 could potentially enable further power saving through more accurate mode configuration in the presented framework. For example, bit-line voltage swing ΔV associated with small signal sensing versus large signal sensing as well as the cost of bit-line/prediction-line switching for each of the considered memory models should be carefully simulated and eventually included in the analysis tools presented in this thesis.

Another potential direction for future work consists in developing a low-power application-specific memory compiler using the results presented in this thesis. The number of prediction and application-to-memory data mapping schemes considered in identifying the most energy-efficient memory configuration for any given data sequence should be expanded. Updating the memory configuration settings at run-time based upon recent data inputs may also prove to be beneficial. However, the consequences associated with the tradeoffs between power consumption, silicon area, and performance need to be carefully analyzed. An additional compiler feature worth exploring is the capability of comparing the performance between a reconfigurable data-dependent SRAM and a conventional SRAM that additionally compresses incoming data based on statistical measures such as correlation. Although compressed data generally requires a smaller memory size and subsequently may save SRAM power and area, the overall system power and area overhead may still be larger due to the requirement of lossless and fast compression and decompression.

Finally, integration of the reconfigurable CP SRAM into larger digital systems or microprocessors is desired to reduce their overall system power consumption.

Bibliography

[1] O. Abari, E. Hamed, H. Hassanieh, A. Agarwal, D. Katabi, A.P. Chandrakasan, and V. Stojanovic. A 0.75-million-point fourier-transform chip for frequency-sparse signals. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2014 IEEE International*, pages 458–459, Feb 2014.

[2] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[3] F. Bossen. Common test conditions and software reference configurations. Document jctvc-h1100, JCT-VC, San Jose, CA, February 2012.

[4] D. C. Brock. *Understanding MooreâĂŹs Law: Four Decades of Innovation*. Chemical Heritage Foundation, 2006.

[5] B.H. Calhoun and A.P. Chandrakasan. A 256kb sub-threshold sram in 65nm cmos. In *Solid-State Circuits Conference, 2006. ISSCC 2006. Digest of Technical Papers. IEEE International*, pages 2592–2601, Feb 2006.

[6] H. Chang, J. Chen, B. Wu, C. Su, J. Wang, and J. Guo. A dynamic quality-adjustable h.264 video encoder for power-aware video applications. *Circuits and Systems for Video Technology, IEEE Transactions on*, 19(12):1739–1754, Dec 2009.

[7] I.J. Chang, D. Mohapatra, and K. Roy. A priority-based 6t/8t hybrid sram architecture for aggressive voltage scaling in video applications. *Circuits and Systems for Video Technology, IEEE Transactions on*, 21(2):101–112, Feb 2011.

[8] H. Fujiwara, K. Nii, H. Noguchi, J. Miyakoshi, Y. Murachi, Y. Morita, H. Kawaguchi, and M. Yoshimoto. Novel video memory reduces 45% of bitline power using majority logic and data-bit reordering. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 16(6):620–627, June 2008.

[9] M. Hahnle, F. Saxen, M. Hisung, U. Brunsmann, and K. Doll. Fpga-based real-time pedestrian detection on high-resolution images. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on*, pages 629–635, June 2013.

[10] H. Hassanieh, P. Indyk, D. Katabi, and E. Price. Simple and practical algorithm for sparse fourier transform. In *Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 1183–1194. SIAM, 2012.

[11] H. Hassanieh, L. Shi, O. Abari, E. Hamed, and D. Katabi. Ghz-wide sensing and decoding using the sparse fourier transform. In *INFOCOM, 2014 Proceedings IEEE*, pages 2256–2264, April 2014.

[12] W. Kong, R. Venkatraman, R. Castagnetti, F. Duan, and S. Ramesh. High-density and high-performance 6t-sram for system-on-chip in 130 nm cmos technology. In *VLSI Technology, 2001. Digest of Technical Papers. 2001 Symposium on*, pages 105–106, June 2001.

[13] T.A. Lahlou and A.V. Oppenheim. Unveiling the tree: A convex framework for sparse problems. In *Proceedings of the 40th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, page unavailable, May 2015.

[14] J. S. Lim. *Two-Dimensional Signal and Image Processing*. Prentice-Hall, 1989.

[15] P. Magarshack, P. Flatresse, and G. Cesana. Utbb fd-soi: A process/design symbiosis for breakthrough energy-efficiency. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2013*, pages 952–957, March 2013.

[16] R. V. Menon, S. Chennupati, N. K. Samala, D. Radhakrishnan, and B. Izadi. Switching activity minimization in combinational logic design. In *nternational Conference on Embedded Systems and Applications*, pages 47–53, June 2004.

[17] K. Mizuno, Y. Terachi, K. Takagi, S. Izumi, H. Kawaguchi, and M. Yoshimoto. Architectural study of hog feature extraction processor for real-time object detection. In *Signal Processing Systems (SiPS), 2012 IEEE Workshop on*, pages 197–202, Oct 2012.

[18] G.E. Moore. Cramming more components onto integrated circuits. *Proceedings of the IEEE*, 86(1):82–85, Jan 1998.

[19] Y. Murachi, T. Matsuno, K. Hamano, J. Miyakoshi, M. Miyama, and M. Yoshimoto. A 95mw mpeg2 mp@hl motion estimation processor core for portable high resolution video application. In *VLSI Circuits, 2005. Digest of Technical Papers. 2005 Symposium on*, pages 212–215, June 2005.

[20] B. Myers, J. Burns, and J. Ratell. Embedded electronics in electro-mechanical systems for automotive applications. *SAE Technical Paper*, page unavailable, 2001.

[21] H. Noguchi, Y. Iguchi, H. Fujiwara, Y. Morita, K. Nii, H. Kawaguchi, and M. Yoshimoto. A 10t non-precharge two-port sram for 74% power reduction in video processing. In *VLSI, 2007. ISVLSI '07. IEEE Computer Society Annual Symposium on*, pages 107–112, March 2007.

[22] J. Oh, G. Kim, I. Hong, J. Park, S. Lee, J. Kim, J. Woo, and H. Yoo. Low-power, real-time object-recognition processors for mobile vision systems. *IEEE Micro*, 32(6):38–50, 2012.

[23] K. Osada, Y. Saitoh, E. Ibe, and K. Ishibashi. 16.7 fa/cell tunnel-leakage-suppressed 16 mb sram for handling cosmic-ray-induced multi-errors. In *Solid-State Circuits Conference, 2003. Digest of Technical Papers. ISSCC. 2003 IEEE International*, pages 302–494 vol.1, Feb 2003.

[24] J.M. Rabaey, A.P. Chandrakasan, and B. Nikolic. *Digital Integrated Circuits*. Pearson Prentice-Hall, 2003.

[25] R. Rithe, C. Cheng, and A.P. Chandrakasan. Quad full-hd transform engine for dual-standard low-power video coding. *Solid-State Circuits, IEEE Journal of*, 47(11):2724–2736, Nov 2012.

[26] M.E. Sinangil and A.P. Chandrakasan. Application-specific sram design using output prediction to reduce bit-line switching activity and statistically gated sense amplifiers for up to 1.9x lower energy/access. *Solid-State Circuits, IEEE Journal of*, 49(1):107–117, Jan 2014.

[27] M.E. Sinangil, A.P. Chandrakasan, V. Sze, and M. Zhou. Hardware-aware motion estimation search algorithm development for high-efficiency video coding (hevc) standard. In *Image Processing (ICIP), 2012 19th IEEE International Conference on*, pages 1529–1532, Sept 2012.

[28] M.E. Sinangil, A.P. Chandrakasan, V. Sze, and M. Zhou. Memory cost vs. coding efficiency trade-offs for hevc motion estimation engine. In *Image Processing (ICIP), 2012 19th IEEE International Conference on*, pages 1533–1536, Sept 2012.

[29] M.E. Sinangil, M. Yip, M. Qazi, R. Rithe, J. Kwong, and A.P. Chandrakasan. Design of low-voltage digital building blocks and adcs for energy-efficient systems. *Circuits and Systems II: Express Briefs, IEEE Transactions on*, 59(9):533–537, Sept 2012.

[30] Y. Sinangil and A.P. Chandrakasan. An embedded energy monitoring circuit for a 128kbit sram with body-biased sense-amplifiers. In *Solid State Circuits Conference (A-SSCC), 2012 IEEE Asian*, pages 69–72, Nov 2012.

[31] J. Singh, S. P. Mohanty, and D. K. Pradhan. *Robust SRAM Designs and Analysis*. Springer Science + Business Media New York, Inc., 2013.

[32] A. Suleiman and V. Sze. Energy-efficient hog-based object detection at 1080hd 60 fps with multi-scale support. In *Signal Processing Systems (SiPS), 2014 IEEE Workshop on*, pages 1–6, Oct 2014.

[33] K. Takagi, K. Mizuno, S. Izumi, H. Kawaguchi, and M. Yoshimoto. A sub-100-milliwatt dual-core hog accelerator vlsi for real-time multiple object detection. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 2533–2537, May 2013.

[34] K. Takeda, Y. Hagihara, Y. Aimoto, M. Nomura, Y. Nakazawa, T. Ishii, and H. Kobatake. A read-static-noise-margin-free sram cell for low-vdd and high-speed applications. *Solid-State Circuits, IEEE Journal of*, 41(1):113–121, Jan 2006.

[35] A. C. Team. *Fundamentals of Global Positioning System Receivers: A Software Approach*. Wiley-Interscience, 2000.

[36] J. Vanne, M. Viitanen, and T.D. Hamalainen. Efficient mode decision schemes for hevc inter prediction. *Circuits and Systems for Video Technology, IEEE Transactions on*, 24(9):1579–1593, Sept 2014.

[37] N. Verma. *Ultra-Low-Power SRAM Design In High Variability Advanced CMOS*. PhD thesis, Massachusetts Institute of Technology, 2009.

[38] Swee Yeow Yap and J.V. McCanny. A vlsi architecture for variable block size video motion estimation. *Circuits and Systems II: Express Briefs, IEEE Transactions on*, 51(7):384–389, July 2004.

[39] K. Zhang, K. Hose, V. De, and B. Senyk. The scaling of data sensing schemes for high speed cache design in sub-0.18 μm technologies. In *VLSI Circuits, 2000. Digest of Technical Papers. 2000 Symposium on*, pages 226–227, June 2000.